



FACULTY OF SCIENCE AND TECHNOLOGY  
MASTER'S THESIS

|  |  |
|--|--|
| Study programme / specialisation:<br>Master of Science in Computer Science   | The <i>spring</i> semester, 2023<br>Open             |
| Author: Ahtsham Shahid, Asim Sabir   |  |
| Supervisor at UiS: Associate Professor Naeem Khademi<br>External supervisor(s): Jens Neesvag (CA&Tech Manager-Tietoevry) |  |
| Thesis title:<br><br>Effective Management of Hybrid Workloads in Public and Private Cloud Platforms                      |  |
| Credits (ECTS): 30   |  |
| Keywords:<br>Hybrid Cloud Management, Hybrid Workload, Unified management, Azure Arc, Multi-Cloud.                       | Pages:74<br>+ appendix:6<br><br>Stavanger, 15.6.2023 |





Faculty of Science and Technology  
Department of Electrical Engineering and Computer Science

# Effective Management of Hybrid Workloads in Public and Private Cloud Platforms

Master Thesis in Computer Science

by

Asim Sabir & Ahtsham Shahid

Supervisor

Assistant Professor Naeem Khademi

Industrial Supervisor

Jens Neesvag (Manager Tietoenvy)



# Abstract

As organizations increasingly adopt hybrid cloud architectures to meet their diverse computing needs, managing workloads across on-premises and on multiple cloud environments has become a critical challenge. This thesis explores the concept of hybrid workload management through the implementation of Azure Arc, a cutting-edge solution offered by Microsoft Azure. The primary objective of this study is to investigate how Azure Arc enables efficient resource utilization and scalability for hybrid workloads. The research methodology involves a comprehensive analysis of the key features and functionalities of Azure Arc, coupled with practical experimentation in a simulated hybrid environment.

The thesis begins by examining the fundamental principles of hybrid cloud computing and the associated workload management challenges. It then introduces Azure Arc as a novel approach that extends Azure control to on-premises and multi-cloud systems. The architecture, components, and integration mechanisms of Azure Arc are presented in detail, highlighting its ability to centralize management, enforce governance policies, and streamline operational tasks. This thesis contributes to the understanding of hybrid workload management by exploring the capabilities of Azure Arc. It provides valuable insights into the benefits of adopting this technology for organizations seeking to optimize resource utilization, streamline operations, and scale their workloads efficiently across on-premises and multi-cloud environments. The research findings serve as a foundation for further advancements in hybrid cloud computing and workload management strategies.



# Acknowledgment

First and foremost, we extend our sincere thanks to our supervisor, Associate Professor Naeem Khademi, from the University of Stavanger. His guidance, expertise, and unwavering support throughout the research process have been invaluable.

We would also like to extend our gratitude to our industrial supervisor, Jens Neesvag, Manager at Tietoevry. His extensive knowledge and expertise in the field have been immensely beneficial to our thesis. We are grateful for his continuous guidance, practical advice, and valuable input, which have greatly contributed to the overall quality of this thesis.

Furthermore, we would like to thank our family and friends for their unwavering support and encouragement throughout this academic journey. Their love, understanding, and patience have been our pillars of strength.

Lastly, we would like to express our gratitude to all the individuals who have directly or indirectly contributed to this thesis. Their valuable feedback, constructive criticism, and suggestions have played a significant role in shaping our research and improving its quality.





# Contents

|  |             |
|--|-------------|
| <b>Abstract .....</b>  | <b>v</b>    |
| <b>Acknowledgment.....</b>   | <b>vii</b>  |
| <b>Abbreviations .....</b>   | <b>xiii</b> |
| <b>1. Introduction.....</b>  | <b>1</b>    |
| 1.1 Motivation.....  | 1           |
| 1.2 Use Cases.....   | 3           |
| 1.3 Problem Definition and Research Questions .....                            | 4           |
| 1.4 Research Objectives.....   | 4           |
| 1.5 Thesis Outline .....   | 5           |
| <b>2. Literature Review .....</b>  | <b>6</b>    |
| 2.1 Introduction.....  | 6           |
| 2.1.1 Overview of hybrid workloads in public and private cloud platforms ..... | 6           |
| 2.1.2 Significance of effective management of hybrid workloads .....           | 8           |
| 2.1.3 Characteristics of cloud Computing.....                                  | 9           |
| 2.1.4 Cloud Computing services models: .....                                   | 9           |
| 2.2 Hybrid Cloud Computing .....   | 11          |
| 2.2.1 Definition of hybrid cloud computing .....                               | 11          |
| 2.2.2 Overview of public cloud platforms .....                                 | 11          |
| 2.2.3 Limitation of public cloud platforms .....                               | 12          |

|           |  |           |
|-----------|--|-----------|
| 2.2.4     | Overview of private cloud platforms .....                        | 13        |
| 2.2.5     | Limitations of private cloud platforms .....                     | 14        |
| 2.2.6     | Overview of hybrid cloud platforms.....                          | 15        |
| 2.2.7     | Limitations of hybrid cloud platforms .....                      | 16        |
| 2.3       | Effective Management of Hybrid Workloads.....                    | 17        |
| 2.3.1     | Definition of workload management .....                          | 17        |
| 2.3.2     | Benefits of effective workload management .....                  | 17        |
| 2.3.3     | Limitations in implementing effective workload management. ....  | 18        |
| 2.3.4     | Challenges and solutions to implementing hybrid workloads: ..... | 19        |
| 2.4       | Conclusion .....   | 20        |
| <b>3.</b> | <b>Hybrid Cloud Methodology .....</b>                            | <b>22</b> |
| 3.1       | Cloud Management Services .....                                  | 22        |
| 3.2       | Why Azure Arc? .....   | 23        |
| 3.3       | Experimental Design.....   | 24        |
| 3.4       | Data Collection and Analysis .....                               | 26        |
| <b>4.</b> | <b>Hybrid Cloud Implementation .....</b>                         | <b>27</b> |
| 4.1       | Experimental Overview .....                                      | 27        |
| 4.2       | Deployment of Azure Arc-enabled Servers- Windows/Linux.....      | 28        |
| 4.3       | Azure Arc-enabled Kubernetes Clusters.....                       | 32        |
| 4.3.1     | On-prem Microk8s Cluster .....                                   | 32        |
| 4.3.2     | Azure Kubernetes Service (AKS).....                              | 41        |
| 4.3.3     | Amazon Elastic Kubernetes Service (EKS).....                     | 45        |
| 4.4       | Deployment of Arc-Enabled SQL Server.....                        | 54        |
| <b>5</b>  | <b>Results and Discussions .....</b>                             | <b>59</b> |
| 5.1       | Introduction.....  | 59        |
| 5.2       | Policies with Azure Arc.....                                     | 61        |
| 5.3       | Security & Governance.....                                       | 62        |

|           |   |           |
|-----------|---|-----------|
| 5.4       | Automation .....  | 63        |
| 5.5       | Seamless Integration and Management .....                 | 65        |
| 5.6       | Capacity for hosting resources.....                       | 66        |
| 5.7       | Cost Efficiency and Utilization.....                      | 67        |
| <b>6</b>  | <b>Conclusion .....</b>                                   | <b>69</b> |
|           | <b>List of Figures .....</b>                              | <b>72</b> |
|           | <b>List of Tables.....</b>                                | <b>74</b> |
| <b>A.</b> | <b>Implementation codes for VMs and SQL servers .....</b> | <b>75</b> |
| A.1       | Onboarding script .....                                   | 75        |
| A.2       | Terraform Code.....                                       | 76        |
| A.3       | SQL installation Script.....                              | 77        |
| A.4       | Deployment.yaml.....                                      | 79        |
| A.5       | Bash Script.....  | 79        |
|           | <b>Bibliography .....</b>                                 | <b>81</b> |



# Abbreviations

|          |  |
|----------|--|
| GCP      | Google Cloud Platform                          |
| AWS      | Amazon Web Services                            |
| API      | Application Programming Interface              |
| NIST     | National Institute of Standards and Technology |
| IaaS     | Infrastructure as a Service                    |
| PaaS     | Platform as a Service                          |
| SaaS     | Software as a Service                          |
| DRaaS    | Disaster Recovery as a Service                 |
| CSP      | Cloud Service Providers                        |
| HCW      | Hybrid Cloud Workload                          |
| CPU      | Central Processing Unit                        |
| EC2      | Amazon Elastic Compute Cloud                   |
| CLI      | Command Line Interface                         |
| k8s      | Kubernetes                                     |
| AKS      | Azure Kubernetes Cluster                       |
| IAM      | Identity Access Management                     |
| SP       | Service Principal                              |
| EKS      | Elastic Kubernetes Cluster                     |
| AAD      | Azure Active Directory                         |
| RBAC     | Role-Based Access Control                      |
| ARM      | Azure Resource Manager                         |
| On-Prem  | On Premises                                    |
| MS Azure | Microsoft Azure                                |



# CHAPTER 1

## 1. Introduction

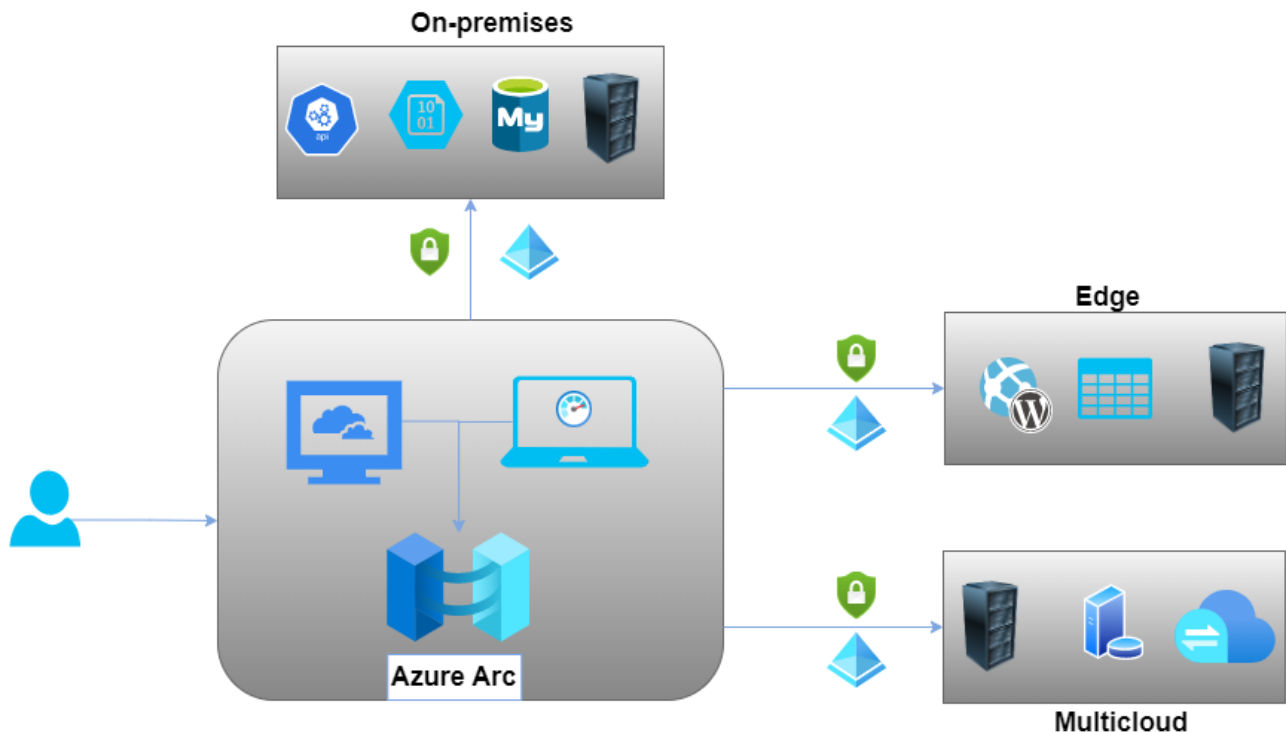
This document details the work done for the Master Thesis titled "Effective Management of Hybrid Workloads in Public and Private Cloud Platforms" for Tietoevry (an IT solution Provider in Norway) during the spring semester of 2023 at the Faculty of Science and Technology at the University of Stavanger (UiS).

### 1.1 Motivation

The expansion of cloud services provided by a wide range of cloud providers from many big tech giants have proven the increased productivity of cloud computing. However, there is still a vast number of end-users who are reluctant to fully adapt to the public cloud due to lack of control, compliances, privacy, and integrity of the data[1]. To address this challenge, hybrid cloud has become a popular choice for organizations looking to balance the benefits of public cloud with the control and security of a private cloud[2]. Along with giving businesses more flexibility and control over their data and applications, hybrid clouds also enable them to adhere to a variety of compliances and standards as well as achieve cost-effectiveness. According to a report from Flexera in 2020, among 750 organizations, 87% of them are using hybrid cloud strategy[3]. There are two types of hybrid clouds strategies. The first type of hybrid strategy (often called multi-Cloud strategy) is when enterprises have resources with a mix of different public cloud like GCP, MS Azure and AWS. The second type of strategy is when the enterprises hold a few resources

on a public cloud and some on-premises. The purpose behind these strategies is to maximize the productivity from different public clouds in terms of specific features and capabilities and to have more control and privacy in case of on-prem. However, managing a hybrid cloud environment can be complex and challenging. Compared to the public cloud, the hybrid cloud has some limitations with integration, security and managing sensitive data[4, 5]. In modern IT, customers need hybrid IT solutions and different platforms have different capabilities for managing resources which makes it more work to maintain and operate.

The motivation behind this research is to meet the consistent growing demand for enterprises, to have one single, flexible, and scalable platform. For this purpose, Tietoevry decided to undertake a thesis on hybrid workload management on private and public cloud with a focus on Azure Arc which is a service provided by Microsoft that enables organizations to manage their IT infrastructure and applications across multiple environments, including on-premises, multi-cloud, and edge, through a unified control plane[6]. For Tietoevry's customers, the goal for



**Figure 1.1:** Hybrid Cloud integration with Azure Arc

having a centralized management solution is to have a seamless experience by reducing the complexity of managing multiple cloud platform, easy integration between on-prem and public



cloud and achieving cost effectiveness while adhering to compliance requirements. Microsoft is continuously improving Azure Arc capabilities to meet the end-user expectations. However, service providers like Microsoft may be subjective to their services to have more sales. Therefore, this thesis aims to describe how to approach an effective and unified way of providing both private cloud and public clouds to end users with the focus on Azure Arc and takes an experimental methodology to analyze the capabilities of Azure Arc in this regard.

## 1.2 Use Cases

There are several use cases of managing hybrid cloud environments with Azure Arc. Companies can administer and control their IT assets deployed anywhere i.e., on public cloud or on-prem. It helps in configuring and enforcing Azure policies for all the resources across any platform. It also enables managing Kubernetes clusters deployed in different environments in a seamless manner[7].

The second use case of Azure Arc is integration with Sovereign Cloud through an API (Application Programming Interface) where enterprises can ensure full data governance and security requirements while complying with regulations and standards. Also, the need for exchanging data between different companies or countries has increased the importance of Sovereign data[8-10]. From Tietoevry standpoint, there has been a lot of work in progress towards Sovereign Cloud due to concerns about data residency, jurisdiction, and data protection.

Another important use case for using this service to manage hybrid cloud environments is that Azure managed identity can easily be integrated with Azure Arc[11]. The administration of secrets, credentials, certificates, and keys used to protect communication between services is a huge challenge for developers. Developers are no longer required to manage these credentials by using managed identities. So, the secure access to resources from different cloud or on-premises platforms can become hassle free as well with Azure Arc by adding an extra layer of security[12].

## 1.3 Problem Definition and Research Questions

With the increasing concerns over data protection, privacy and security, organizations move to Hybrid cloud approach, but it comes with much more complex challenges i.e., ineffective control over resources, lack of competence and security vulnerabilities that may lead to additional cost[1, 2]. These challenges can be addressed by different solutions by different cloud providers such as AWS outpost, IBM Park, Microsoft Azure Arc etc.[6, 13, 14] Tietoevry provides cloud and support services, mainly Microsoft Azure, to many of its clients for their cloud or on-prem infrastructure. However, for managing hybrid environments using these solutions may lead to more complexity and inconsistencies if not implemented correctly.

This work scrutinizes Azure Arc to provide a centralized platform and analyzes its potential for the level of control it offers and identify the best implementation practices. With comprehensive understanding of Azure Arc, Tietoevry will be able to make an informed decision on whether to adopt Azure Arc for hybrid cloud management needs. This leads to the following research questions:

- *RQ1*: What are the best practices and challenges of using Azure Arc for hybrid cloud management?
- *RQ2*: How does the use of Azure Arc and its related services affect the security of a hybrid cloud environment and what level of control the end-user achieves with Azure Arc?

These research questions will be further explained in the later part of the thesis.

## 1.4 Research Objectives

A list of main objectives can be identified by defining the main research questions:

1. Study the hybrid cloud challenges and previous work for mitigation the challenges and investigate Azure Arc and its best practices to manage hybrid environments.

2. Define a plan for Infrastructure for on-prem and public cloud with multi-cloud approach and deploy and test the plan.
3. Set up the integration of the deployed resources with Azure Arc and investigate security and access to the resources.
4. Obtain feasibility results for the deployed infrastructure and observe the capabilities of Azure Arc in the light of the obtained results.

## 1.5 Thesis Outline

This document has been organized as follow:

- Chapter 2 describes the theory behind Hybrid Cloud management and relevant work done on it, adding some insights into the challenges of adopting Hybrid Cloud management and some unified planes to address these challenges.
- Chapter 3 includes a brief explanation of the research methodologies, adding insights into reasons for choosing the experimental tool and mapping of the infrastructure for experiments.
- Chapter 4 discusses in profound detail the resources used and implementation of the plan from chapter 3.
- Chapter 5 comprises of implementation procedure and results from it.
- Chapter 6 details the results, analyses of the outcomes, and investigates limitations of Azure Arc.
- Chapter 7 summarizes the outcome of this research and points out the factors for choosing the experimented solution.

# CHAPTER 2

## 2. Literature Review

### 2.1 Introduction

#### 2.1.1 Overview of hybrid workloads in public and private cloud platforms

Hybrid workloads are deployed across both public and private cloud platforms. This allows the organizations to get the benefit of both public and private cloud platforms that is ideal for their business requirements. In hybrid mode, some of the workloads are deployed on private clouds and some are deployed on public clouds[\[15\]](#).

Public clouds are hosted by third-party service providers, some of them are AWS by Amazon, Microsoft Azure by Microsoft, and GCP by Google[\[16\]](#). These providers offer many advantages and organizations use these advantages based on their needs. Some important advantages are:

- Scalability
- Cost-Effectiveness
- Ease-of-use

On the other hand, private cloud platforms are hosted inside the organization's own infrastructure or data centers[16]. Private cloud platforms are built using VMware, OpenStack, and Microsoft Azure Stack. Some benefits provided by private cloud platforms include:

- Security
- Control
- Compliance

Companies use hybrid cloud platforms when they need the advantage of both cloud platforms. For example, some of that data that is private to the organizations i.e., employees or customers data, can be stored on a private cloud platform. The remaining data can be stored on public cloud platforms to take advantage of scalability and cost-effectiveness.

Organizations use various tools to deploy hybrid workloads, that includes cloud management platforms, containerization, and automation tools. These tools help in managing data across public and private clouds and in return provide a smooth experience for the customers.

In summary, hybrid clouds are becoming popular, as they let the organization take benefit of both public and private clouds. This also helps them achieve better efficiency, scalability, and security. Table 2.1 shows the comparison of different cloud computing platforms.

Table 2.1: Different cloud computing models and their comparison [15]

| Model   | Hosted by            | Security                            | Scalability | Cost |
|---------|----------------------|-------------------------------------|-------------|------|
| Private | private organization | Higher than other deployment models | Limited     | High |

|        |                               |   |           |             |
|--------|-------------------------------|---|-----------|-------------|
| Public | Service Provider              | Lower than other deployment models                    | Very High | Pay-per-use |
| Hybrid | CSP and private organizations | Higher than Public Cloud and lower than Private Cloud | High      | Pay-per-use |

### 2.1.2 Significance of effective management of hybrid workloads

There are many reasons why an organization should choose hybrid workload management. Some of the reasons are:

1. **Cost Optimization:** Organizations can manage their IT costs by choosing cost-effective platforms for their specific workloads. It helps them use their resources efficiently while monitoring and controlling their costs [17].
2. **Scalability:** Depending on the changing needs, organizations can scale up their resources to meet the requirements of increasing customers on their platforms, without needing to install new hardware every time they need to scale up [18]. Organizations can even scale down to decrease costs or because they have more resources than they require.
3. **Security:** Organizations can manage their data on public and private platforms. By effective management of public and private platforms companies can ensure their data is secure and protected from cyber threats [19].
4. **Agility:** Hybrid workloads help organizations to be more agile according to the changing business needs by automating the workflows.

Effective management is essential for organizations if they need to adopt hybrid cloud workloads management. This benefits organizations in the long term while keeping business costs low.

### 2.1.3 Characteristics of cloud Computing

There are several characteristics of cloud computing that are provided by the National Institute of Standards and Technology (NIST)[\[20\]](#). These characteristics are:

1. On-demand self-service: The process of accessing cloud computing applications is automated and does not require any human input. This means that a user can access cloud computing services quickly and easily, without any need to interact with the cloud service provider.
2. Broad network access: The services provided by the cloud computing platforms are easily accessible and can be operated from various devices, for example tablets, laptops, mobile phones and workstations.
3. Resource pooling: Cloud computing providers use multiple sources to provide computational power to the users. This model is called multi-tenant model. According to the consumer demand, different resources are assigned and reassigned. This is solely done by the service providers and the customers have no control over the location of the resources. Although they can specify the location at a higher level.
4. Rapid elasticity: Consumers can easily scale up and scale down resources based on their own needs. They can automate the utilization of as many resources as possible.
5. Measured service: Cloud computing service providers provide complete transparency and show the usage of the resources. This helps the customers understand that they are using all the resources and if this is not the case they can scale down.

### 2.1.4 Cloud Computing services models:

IaaS (Infrastructure as a Service) architecture provides computer sources to the users. These sources are virtualized and include operating systems, memory, processing power, and application software. These sources are logical sources and can be provisioned and released based on customer demand. Some of the IaaS service providers are Amazon EC2, Google, Verizon, IBM,

and Rackspace cloud servers. IaaS provides several benefits to its users which include reducing the cost by pay-per-use policy, providing high-quality resources and infrastructure, and scaling up and down the resources based on the user's demand to save cost or time.

PaaS (Platform as a Service) architecture helps the user to manage their software, operating systems, and computing resources [21]. This allows the users to share their applications among others. Services provided by PaaS included designing, developing, and hosting applications. Other than that, it helps in collaborating web service integration, security, database integration, and scaling. Users buy access to the platform, to deploy their applications and software on the cloud. Some of the PaaS providers are Google App Engine, Salesforce, Microsoft Azure, and Rackspace Cloud Sites. PaaS provides a platform for building and supporting a strong cloud app development community. This is done by avoiding the need for upgrades as the patches, upgrades, and maintenance are handled by the service provider. PaaS has many disadvantages too, that includes portability, and interoperability among providers.

SaaS (Software as a Service) architecture provides the service of running and maintaining the operating systems, application software, and other resources to its users[21]. This model provides a web-based application that can be accessed by customers through any web browser. This saves the hassle of buying the license, installing, upgrading, maintaining, and running the application on the customer's computer. Many other benefits provided by SaaS included configurability, scalability, and multitenant efficiency. Other than that, accessibility from any location with an internet connection, rapid scalability, elimination of infrastructure concerns, elimination of infrastructure concerns, and custom levels of service offerings are also the benefits of SaaS.

Recovery as a Service (RaaS), also known as DRaaS (Disaster Recovery as a Service) architecture provides organizations with solutions to replace their disaster recovery, archiving, backup, and business continuity solutions. Raas can help companies to recover their database files, servers, and data centers easily. This results in reduced downtime in case of any emergency. Some main RaaS providers are Geminare, and WindStream Business. Raas can prevent companies from temporarily or in the worst-case scenario permanently losing their data, and



physical infrastructure. This all is done by providing cost-efficient service while maintaining the flexibility and accuracy in the type of backup required[22].

## 2.2 Hybrid Cloud Computing

### 2.2.1 Definition of hybrid cloud computing

Hybrid cloud computing refers to the environment that combines two or more cloud computing platforms[16]. Usually, public and private clouds are combined for this purpose. This computing model lets the organizations share the data and applications between public and private clouds. By doing so the organizations get the benefits of both public clouds which are low cost, and flexibility, and the benefits of private clouds which are security and more control.

Public clouds can be used to manage high traffic on the website or large volumes of data. While private clouds can be used to store user data that is sensitive or data that requires high security and compliance.

### 2.2.2 Overview of public cloud platforms

Public cloud platforms are services provided by third-party providers and are available to users over the Internet. These cloud platforms provide computing resources such as storage, networking, databases, and applications. This helps organizations to scale their resources without the use of any hardware. Some of the most popular cloud platforms are [23]:

1. Amazon web services (AWS) [24] is the most popular and widely used public cloud platform. It not only offers typical cloud services such as computing, storage, networking, and databases but also tools for machine learning, analytics, and the Internet of Things (IoT).
2. Microsoft Azure[25] is a cloud platform offered by Microsoft. In addition to usual cloud services, it also provides integration with other Microsoft platforms such as Office 365, and Dynamics 365.

3. Google Cloud Platform (GCP) [26] is a cloud platform offered by Google. Along with normal cloud services, it also provides Data Analytics and IoT services.
4. IBM Cloud [14] is a public cloud platform by IBM. They provide typical cloud services and provide tools for blockchain and IoT.
5. Oracle Cloud [27] is a public cloud platform offered by Oracle. In addition to usual cloud services, they also provide tools for artificial intelligence (AI), analytics and IoT.

Public cloud platforms provide various benefits including scalability, flexibility, cost saving, and ease of use. They are an important part of every organization from startups to large enterprises.

### 2.2.3 Limitation of public cloud platforms

There are many limitations of public cloud platforms that organizations suffer from. One of the major limitations is control over the infrastructure or the cloud. Public cloud providers are the ones in control, so the organization relies on them for security, updates, and maintenance.

Another limitation is the potential vendor lock-in[28]. This means that a specific public cloud service provider gives specific applications and APIs which are only available on their cloud[28]. So, if the public cloud provider changes the policies or costs which do not sit right with the organization, it becomes difficult to switch to other public clouds.

Public clouds sometimes suffer from performance issues[28]. As public clouds share the resources, sometimes performance issues can occur during the peak usage period.

One other major concern is the security of the data [29-31]. Organizations rely on public service providers to implement appropriate security measures to make their customer's data safe. Public clouds sometimes suffer from data breaches, which can lead to serious consequences for the organizations and may cause legal issues for them.

Finally, public clouds can become expensive for organizations that need high computational power and storage. Public cloud providers usually charge based on pay-per-use, which can cause unpredictable costs.

#### 2.2.4 Overview of private cloud platforms

Private cloud platforms are cloud computing services that are dedicated to one organization. They have the same benefits as public clouds such as scalability, flexibility, and automation. Although to scale up the system new hardware is required and when infrastructure is scaled down, the extra hardware remains unused. Public cloud platforms do not share their infrastructure with any other organization; thus, they provide more security and control. Some popular private cloud platforms are:

1. VMware vSphere[32] is a popular private cloud platform that provides storage, network resources, and a virtualization environment for computing. It helps organizations to create and manage private cloud platforms easily. They offer high availability, disaster recovery, and automation.
2. OpenStack[33] is an open-source cloud platform that helps in building private cloud platforms that are scalable and flexible. It provides features such as storage, networking, computation, and security services.
3. Microsoft Azure Stack [34] is a private cloud platform provided by Microsoft. It provides a consistent platform between the cloud environment and the On-premises environment. Services provided by this platform are storage, Networking, Computation, and security. In addition to that, it also provides integration with other Microsoft services such as Office 365 and Dynamics 365.
4. IBM Cloud Private [35] is a private cloud platform by IBM. It acts as a container for building and running cloud-native applications. In addition to providing services like storage, networking, computation, and security services, it also supports different container-based platforms.
5. Nutanix Enterprise Cloud [36] is a private cloud platform that provides hyper-convergence infrastructure for building and managing private clouds. It provides services like storage, networking, computation, and security services.

Private cloud platforms offer many benefits such as security, control, and customization. They are used by organizations that require full control over their IT infrastructure.

### 2.2.5 Limitations of private cloud platforms

Private cloud platforms have several benefits, but they also come with several limitations [37].

Here are some common limitations of private cloud platforms:

1. **High-Cost:** One of the most significant limitations of private cloud platforms is the high cost of building and maintaining the infrastructure. Private clouds need a lot of investment in terms of hardware, software, and other infrastructure components. Also, private cloud infrastructure requires ongoing maintenance like upgrades and patches, which can cause high maintenance costs.
2. **Limited Scalability:** Private cloud platforms are not easily scalable when compared to public cloud platforms. If resources are not being used to the fullest, then the extra hardware remains idle, and if the hardware does not meet the needs of the organizations, then extra hardware is required to scale up.
3. **Complexity:** Private cloud platforms are extremely complex to deploy and manage. They require individuals who have expertise in networking, storage, and security. As a result, private cloud platforms are not recommended for small organizations that lack the resources to manage the private cloud infrastructure.
4. **Security:** Although private clouds provide higher security than public clouds, they can also pose a threat to security. They need careful management in terms of control and security. Any vulnerability can result in significant consequences.
5. **Limited access:** Private clouds are only used inside the organization and that limits its access to authorized users. This can cause a disadvantage to an organization that needs to collaborate with partners or customers from outside the organization.

Private clouds have several limitations in terms of cost and managing them. Organizations need to consider these threats when implementing private cloud infrastructure.

### 2.2.6 Overview of hybrid cloud platforms

Hybrid cloud platforms combine the benefits of both public and private cloud platforms[15]. It is a unified infrastructure that allows organizations to combine public and private clouds based on the organization's needs and requirements. Some popular hybrid cloud platforms are:

1. Microsoft Azure Stack Hub [38] is a hybrid cloud platform by Microsoft that allows organizations to use a consistent platform between the cloud environment and the On-premises environment. It provides unified management and seamless integration between both environments.
2. AWS Outposts [13] is a hybrid cloud platform by AWS. It allows organizations to use AWS services in their own data centers. It provides fully managed services that enable organizations to run workloads in the cloud and on-premises.
3. Google Anthos [39] is a hybrid cloud platform that helps organizations to build and manage applications on multiple cloud platforms including public clouds, private clouds, and on-premises data centers. It helps organizations manage their workloads and to move them across different environments.
4. IBM Cloud Satellite [40] is a hybrid cloud platform that helps in managing workloads across different clouds, data centers, and edge locations. It provides unified management to help organizations deploy and manage their applications across different platforms.
5. VMware Cloud [41] on AWS is a hybrid cloud platform that allows organizations to run VMware workloads on AWS infrastructure. It is a fully managed service that can be integrated with AWS and helps the organization to run workloads on-premises and in the cloud.

Hybrid clouds offer benefits like flexibility, scalability, and security, on top of all they are cost-efficient. They help organizations acquire the benefits of both cloud environments and help them manage their workload appropriately.

### 2.2.7 Limitations of hybrid cloud platforms

In recent years, hybrid cloud platforms have gained popularity as they provide benefits to both public and private cloud platforms and cover the limitations of both cloud platforms to some extent. However, they also have limitations [42], some of those limitations are:

1. Complexity: Hybrid cloud platforms are complex to manage as they have multiple components and needs integration between these components[43]. Specific skills and expertise are required to manage hybrid clouds effectively.
2. Cost: Hybrid cloud platforms are expensive to implement and maintain. Organizations bear the cost of both public and private cloud infrastructures which can add up quickly. This is why a hybrid cloud platform is not recommended for smaller organizations [44].
3. Security: Security is one of the significant concerns for a hybrid cloud environment [45]. Due to the higher complexity of the infrastructure, it becomes challenging to ensure the security of data and there is a risk of cyber-attacks.
4. Data management: It is challenging to manage data in multiple environments. It requires careful planning, coordination, and management to ensure that the data is only available where it is needed and when it is needed.
5. Latency: Latency is also an issue with hybrid clouds. As hybrid clouds are built on multiple systems, it becomes difficult to make data and applications available quickly.
6. Dependence on internet connectivity: Hybrid clouds rely heavily on good internet connectivity which can be a limitation for organizations with poor and unreliable connectivity. This can affect the availability of applications and data which can decrease productivity.

Overall, hybrid clouds may provide several benefits for both public and private cloud platforms, organizations should consider these limitations and ensure they have the proper resources to implement hybrid cloud infrastructure.

## 2.3 Effective Management of Hybrid Workloads

### 2.3.1 Definition of workload management

Workload management is the process of planning, organizing, and optimizing the distribution of tasks, processes, and resources on an organization's computing platforms. By managing the workload, the organization ensures that all the resources are being used in an efficient and effective way. This saves them both cost and time.

Workload management usually revolves around monitoring and analyzing the performance of all the computing resources, including storage devices, servers, and network resources. Workload management can be applied to public, private, or hybrid computing infrastructure. It lies under the umbrella of IT management to achieve optimal performance, reduce costs, and improve productivity.

### 2.3.2 Benefits of effective workload management

Effective workload management can benefit an organization on many levels[46]. It can help increase efficiency and thus productivity. Some of the advantages are:

1. Improved Performance: Effective workload management ensures that the organization is running in an optimal way by prioritizing and optimizing the workloads [47]. This results in increased performance, reduced downtime, and faster response time.
2. Increases efficiency: Organizations can reduce wasted resources and improve utilization rates through effective workload management [48]. This not only increases efficiency but also helps in reducing the cost.

3. Greater flexibility: Effective workload management can help in shifting resources and workloads as necessary to meet the changing business needs [49]. This can help the organization to quickly adapt to the changing market, customer demands, and other challenges.
4. Enhanced Security: Workload management can help in increasing security and ensures that the workloads are secure, compliant, and protected from cyber-attacks[50]. Organizations can reduce the risk of data breaches and other security incidents, by properly securing and managing the workloads.
5. Improved scalability: Effective workload management can help organizations to scale their resources as needed to meet the changing demands. This can help organizations to ensure that they have the resources that they need to support their business growth.

### 2.3.3 Limitations in implementing effective workload management.

Although effective workload management has many benefits, it also has several limitations [37]. These limitations mostly occur due to limited resources. Some of these limitations are:

1. Resource constraints: Effective workload management needs sufficient resources including computing power, storage, and internet connectivity. If an organization fails to acquire these resources or if these resources are not managed efficiently, then workload management can become difficult.
2. Technical Compatibility: Sometimes, the tools and software to be used for workload management are not compatible with the existing IT infrastructure of the organization. This creates many technical challenges and limits the effectiveness of workload management.
3. Resistant to change: It is noticed in many organizations that employees are reluctant to change and adopt new workload management tools and applications. This can hinder and slow down the implementation and adoption of new workload management strategies.



4. Lack of expertise: Effective workload management requires a workforce that is highly skilled in the relevant technologies and processes. If an organization fails to acquire the experts, then it becomes difficult to manage the workload effectively.
5. Cost: There are many tools and technologies that an organization might need to invest in to implement effective workload management strategies. Based on the organization's budget, this may be a limiting factor.

2.3.4 Challenges and solutions to implementing hybrid workloads:

There are several challenges associated with implementing effective workload management [37]. Some of the challenges and solutions are given in table 2.2.

Table 2.2: Challenges and solutions associated with effective workload management in hybrid platforms.

| Challenge  |  | Solution       |  |
|------------|--|----------------|--|
| Complexity | A hybrid cloud platform consists of multiple environments. It makes it difficult to keep consistency and compatibility between different systems while managing workloads. | Automation     | Automating the process of workload management can help in reducing the complexity and ensuring consistency across different platforms. |
| Security   | There are many unique security challenges related to hybrid cloud platforms. These challenges include data   | Security tools | To ensure the transmission of data between different environments, firewalls, and encryption can be implemented.                       |

|                     |  |                           |   |
|---------------------|--|---------------------------|---|
|                     | transmission between different environments and maintaining consistent security policies across different environments.  |                           |   |
| Resource allocation | Managing resources across different environments can be challenging as it is difficult to allocate resources optimally and to ensure that all the workload is properly distributed across different platforms. | Resource management tools | Implementing resource balancing and auto-scaling can help in optimizing the allocation and workload balancing across different platforms.                       |
| Cost                | A significant investment in infrastructure, staff training, and ongoing maintenance is required for managing hybrid workload management which can be expensive.  | Cost optimization         | The use of reserved instances and leveraging stop instances can help in reducing the cost associated with workload management across different cloud platforms. |

## 2.4 Conclusion

The literature review covered the types of cloud platforms and which service providers are there to build the infrastructure of different cloud platforms including Microsoft Azure Stack Hub, AWS Outposts, Google Anthos, IBM Cloud Satellite, and VMware Cloud on AWS. The review also discussed the benefits of using each type of cloud infrastructure i.e., public, private, and hybrid. While the literature review gave an overview of hybrid cloud platforms and their benefits,

there are many limitations in terms of in-depth analysis of specific features and functionalities. Additionally, potential risks and challenges are associated with implementing these platforms in real world scenarios.

# CHAPTER 3

## 3. Hybrid Cloud Methodology

This chapter lists the key functionalities required by a tool to monitor and manage cloud resources. It involves different approaches and tools used to address the problem and compare them to find the best results. Which tool we used for this work and why. Planning and assessment of the labs and way to carry out the data collection and analysis.

### 3.1 Cloud Management Services

Different cloud providers offer different management services to the customers to manage their hybrid workloads in terms of monitoring, managing cloud environments (Production, Testing, Development), and scaling the resources. These management services are used to manage resources for different cloud types such as Private, Public, Multi-Cloud and Hybrid Cloud. These services provide several technical capabilities, which include orchestration, optimization, security, and data monitoring. Such services also offer open-source software modules to support public and private cloud environments. Selecting such services by enterprises depends on the following factors.

1. If the organizations are using different cloud platforms to deploy their resources. And there is a need for organizations to monitor and optimize cloud resources from different cloud platforms.

2. Organizations are concerned about transparency, insights, analytics, and governance for their resources. So, having the resources in a single platform will help the organizations to manage the resources in a more efficient way.

In general, there are multiple tools available in the market for unified monitoring. Choosing one of them depends on the customer and their needs. Later we discuss which tool we choose in this work and why. Choosing the best tool consists of the different scenarios such as ease to adopt for an organization, level of the support available in case of help required and product reputation in the market. The following are a few points to be considered while choosing the most suitable tool for an organization[51].

1. Business Size: The size of an organization helps them to choose the right tool for their need. Multiple business sizes (Small, Medium and Enterprise) have different tools. For example, Small Business we have (Google Compute Engine, Utho, nOps, DoiT). For Medium Business we have same as small but some additional (ClouKeeper, IBM Cloud Pak for AIOps, OpenStack etc.). For Enterprise businesses we have the same as small and medium but with more tools like (Azure Arc, Turbonomic, BMC Software).
2. Rating: Rating for different tools describes which one is best among others. Tools with higher ratings means it has been used by multiple organizations and provides more reliable results.
3. Pricing: Either the tools are free to use or there is a cost to implement and manage the resources from them. Mostly the tools are free to onboard the machines but there is cost included when to extend the functionality of the tool.

## 3.2 Why Azure Arc?

This work is done for Tietoevry, which is a Microsoft Partner and the tool we choose for unified monitoring is Azure Arc. The main reason for Azure Arc is that most of the Tietoevry customers are using Microsoft Azure for their resources and deployments. Azure Arc is a Microsoft management tool which helps in unified monitoring as a centralized platform. Tietoevry employees are more confident to use this tool as Microsoft Azure is widely used in the company

as well as in Norway. One more reason to choose azure arc is that our departments wanted to implement unified. As Tietoevry recently introduced the sovereign cloud for the customers, this tool can be beneficial as well in distinct aspects like data governance, data integrity, monitoring, and insights etc.[\[52\]](#).

### 3.3 Assessment and planning

This stage involves evaluating the existing infrastructure and identifying the resources to be managed with this tool. These experiments are performed by using Microsoft Azure portal given by Tietoevry to explore azure arc. Machines are onboarded from different platforms to azure portal. We created a student account on AWS to create instances for free of cost and later onboarded to our azure subscription. Lab is also designed with on-premises infrastructure onboarding with over two servers, including (Microsoft Server, Ubuntu Server). We deployed virtual machines on our local systems using VMware and onboarded both on azure portal by using azure arc agent installing on both systems. In AWS, we deployed one normal virtual machine to implement azure arc enabled server and one virtual machine with SQL services to implement Azure Arc enabled SQL server.

### 3.3 Experimental Design

Different scenarios have been considered for experiments with the Azure Arc by implementing resources and services such as Subscriptions, Resources Groups, Virtual Machines, Networks, Gateways, Kubernetes Clusters. In general, lab design with Azure Arc involves below steps[\[53\]](#):

1. Setting up the subscriptions: Two different subscriptions have been set up to onboard the servers and clusters. The purpose for different subscription is because of using it from two different departments inside TietoEVRY.
2. Resource Groups: Two different resource groups are created both serve as the central hub for resources and used to connect servers to it. For Arc Enabled servers and SQL resource group (HCW-RG) has been used. For Kubernetes cluster resource group (rg-Arc) has been used.

3. Adding Infrastructure: From azure arc portal we added the infrastructure and generated the script to run on our different platforms for onboarding purposes. Fetch the scripts from Azure Arc for respective resource and install the Azure Arc agent on machines on-premises, multi-cloud, or edge servers by running this script.
4. Monitor Resources: Monitoring the health and performance of Azure Arc enabled servers using Azure Monitor, Log Analytics, and Azure Resource Graph[54].
5. Implement Azure Policies: Implementing Azure policies to enforce compliance and governance for servers and applications.
6. Update Servers: Using Azure Arc to apply updates and patches to servers, ensuring they remain secure and up to date.

These steps are involved in Azure Arc lab environment, providing unified management experience for multi-cloud, on-premises, and edge resources. Also, by offering several features and capabilities that assist enterprises in adhering to their regulatory and legal obligations for data security and privacy, and data compliance[55]. A brief description of these features is discussed below.

1. Data Encryption: To safeguard sensitive information from unauthorized access, Azure Arc offers data encryption both at rest and in transit.
2. Role-Based Access Control: With the aid of role-based access control in Azure Arc, administrators may implement the least privilege principle by limiting who has access to confidential information and resources.
3. Auditing and Reporting: Azure Arc has thorough auditing and reporting features that let businesses keep track of who has access to sensitive information and make sure they are following rules.
4. Data Residency: By enabling businesses to store their data in particular geographical regions, Azure Arc meets data residency requirements by ensuring that the data is kept under the organizations' control and complies with local laws and regulations.

Once the desired results have been obtained, they need to be completely studied and interpreted. This involves analyzing if the results match the expectations and finding out the unexpected

behavior of the results and implementation of the given data and scenarios formulate a series of conclusions and answers to the research questions.

### 3.4 Data Collection and Analysis

Azure Arc gathers a variety of data types to offer metrics, analytics, and insights that assist businesses in managing their workloads and resources. Some of the data that Azure Arc may collect includes: [\[54\]](#)

- Resource metadata: Information about the resources and nodes managed by Azure Arc, such as names, IP (Internet Protocol) addresses, and operating system versions.
- Performance metrics: Data about resource utilization, such as CPU (Central Processing Unit) utilization, memory usage, and network traffic.
- Configuration data: Information about the configuration of resources and nodes, including installed software, settings, and policies.
- Log data: Detailed log entries that provide insights into the operation of resources and nodes, including system events, error messages, and performance information.
- Compliance data: Information about the compliance status of resources and nodes, including the results of security scans, vulnerability assessments, and compliance audits.
- User activity data: Details about user actions, such as who performed a certain action, when it was performed, and from which IP address.

Azure Arc uses this data for reporting and analyses. This data can be used to keep a watch over resource performance, identify issues, and to make sure that it is compliant with policies and regulations. The collected data can also be utilized for other Azure services, such as Azure Monitor and Azure Security Center, to provide a comprehensive view of resource and workload management.



# CHAPTER 4

## 4. Hybrid Cloud Implementation

This chapter described the implementation scenarios of Azure Arc, by implementing different azure arc-oriented labs. Labs are based on On-premises, multi-cloud environments.

### 4.1 Experimental Overview

Main goal of the experimental overview is to provide a working azure arc environment to focus on the values of different platforms no matter from where the data is being retrieved. We define various types of server distribution of Azure Arc-enabled server which consists of Windows and Linux based distributions i.e., VMware vSphere resource management with azure arc, Onboarding Microsoft SQL Server as an Azure Arc-enabled SQL Server, Kubernetes clusters distribution. Deploying and managing the Azure Arc-enabled data services on multiple infrastructure platforms. Machine Learning scenarios on Kubernetes manage and deploy by Azure Arc. In the end we describe the central operations carried out by Azure Arc and Azure Lighthouse. To perform the experiments, we have two Microsoft Azure portal subscriptions where the actual work happens like deployment, monitoring etc. Two different Amazon Web Services and Microsoft Azure accounts to deploy the resources and later integrate with Azure Arc which serves the purpose of Multi-cloud Environment. To integrate with on-premises load, few servers are deployed locally on local machines in VMware and few servers are connected from the data centers. Every single step is included to get this work done later in this chapter.

## 4.2 Deployment of Azure Arc-enabled Servers- Windows/Linux

In this section, existing windows and Linux servers are onboarded from different cloud providers.

First part we focus on AWS instance, The deployment of an AWS instance to Azure Arc involves registering the instance with Azure Arc and installing the Azure Arc agent on the instance. This enables us to manage and monitor the instance through the Azure Arc control plane. Once the agent is installed and the instance is registered, we can apply policies, view inventory, and monitor the instance's health and performance. This integration between AWS and Azure Arc allows us to manage resources across multiple clouds from a single control plane.

To achieve this, a dummy account is created with student subscription. The account provides access to various AWS services, such as EC2, S3, and RDS, as well as free credits to use these services. With a student account, users can practice hands-on with AWS services, build projects, and develop skills in cloud computing. However, the student account has some limitations, such as a spending limit on credits and restricted access to certain features. To create a student account, users need to provide their academic information and a valid credit card to verify their identity.

In Azure Arc module, three different options are available to deploy the resources.

1. Add a Single Server
2. Add Multiple Sever
3. Add Servers from update Management.

Our focus for the multi-cloud environment is to choose a single server for testing, the first option will generate a script to run on our targeted server.

Pre-Requisites:

To connect an AWS EC2 instance to Azure Arc, following prerequisites:

1. An active Azure subscription: We needed an Azure subscription to use Azure Arc.
2. Azure Arc enabled servers: AWS EC2 instances must have the Azure Arc agent installed on them.
3. Access to Azure Arc enabled servers: Requires administrator access to the AWS EC2 instances to install the Azure Arc agent and configure them to connect to Azure Arc.
4. Azure Arc enabled Kubernetes clusters: If we want to connect AWS EC2 instances running Kubernetes to Azure Arc, you must have an Azure Arc enabled Kubernetes cluster.
5. Azure Arc enabled data services: If we want to connect AWS EC2 instances running SQL Server or PostgreSQL to Azure Arc, must have an Azure Arc enabled data service.
6. Permissions: Necessary permissions to perform the required actions in Azure, AWS, and on the AWS EC2 instances.

Here are the high-level steps to connect an AWS EC2 instance to Azure Arc:

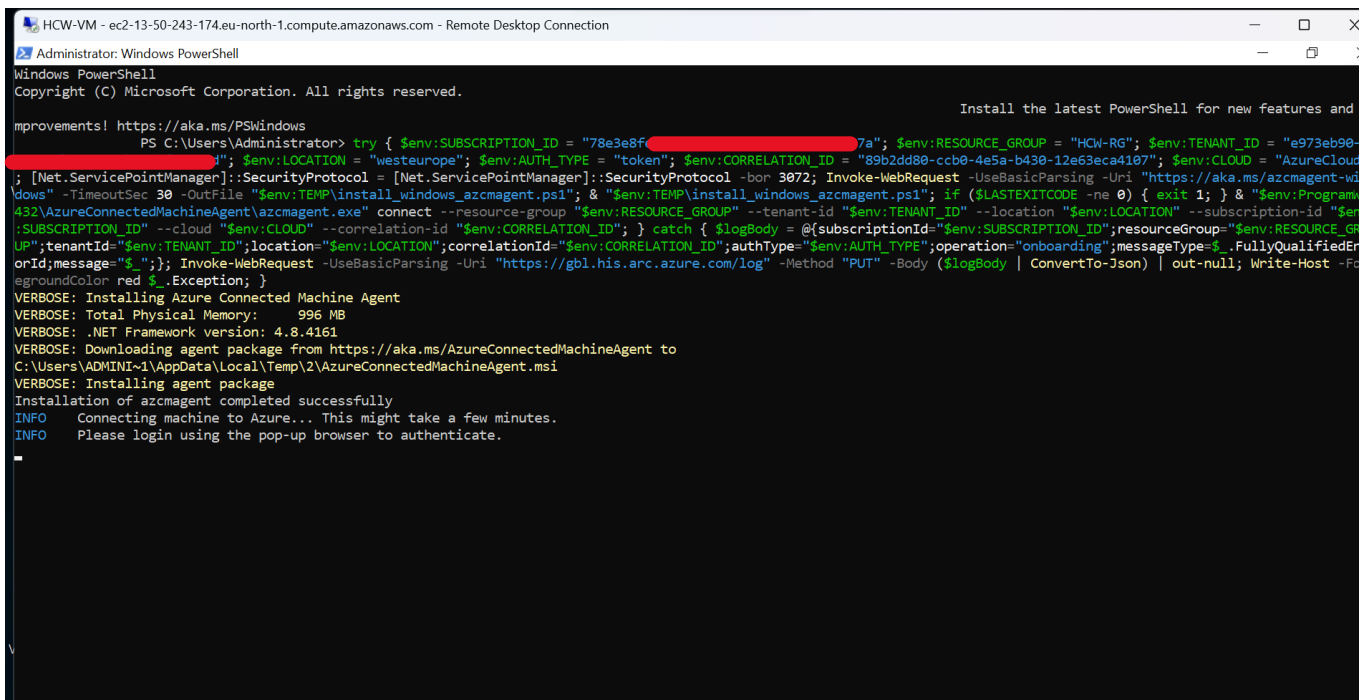
1. Enable Azure Arc for servers and created a resource group called HCW-RG in Azure.
2. Create an onboard script from azure arc module and past it to the AWS instance PowerShell.
3. Install the Azure Arc agent on the AWS EC2 instance.
4. Register the AWS EC2 instance with Azure Arc using the Azure Arc agent.
5. Verify that the AWS EC2 instance is connected to Azure Arc.

Here are the detailed steps:

Enable Azure Arc for servers and created a resource group called HCW-RG in Azure:

- a. In the Azure portal, search for and select "Azure Arc".
- b. Select "Servers" from the left-hand menu, and then select "Azure Arc enabled servers".
- c. Follow the prompts to enable Azure Arc for servers, select your Azure subscription, and create a new resource group if necessary.
- d. Created a service principal in Azure to provide Azure Arc with access to your AWS account.

1. Log in to the AWS EC2 instance and download the Azure Arc agent from the Azure portal.
  - a. Install the agent on the AWS EC2 instance by following the instructions provided by Azure.
  - b. Create an onboard script from azure arc module and past it to the AWS instance Powershell.
    - i. The onboarded script used for onboarding of the AWS instance which installs the azure arc agent on Windows Server can be found in Appendix [A.1 Onboarding script](#). [56]
2. Register the AWS EC2 instance with Azure Arc using the Azure Arc agent:
  - a. On the AWS EC2 instance, run the Azure Arc agent and follow the prompts to register the instance with Azure Arc.
    - i. Here is the picture attached onboarding of an AWS Windows Machine to Azure Arc.



```

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and
improvements! https://aka.ms/PSWindows
PS C:\Users\Administrator> try { $env:SUBSCRIPTION_ID = "78e3e8f-7a"; $env:RESOURCE_GROUP = "HCN-RG"; $env:TENANT_ID = "e973eb90-
"; $env:LOCATION = "westeurope"; $env:AUTH_TYPE = "token"; $env:CORRELATION_ID = "89b2dd80-cb9-4e5a-b430-12e63eca4107"; $env:CLOUD = "AzureCloud
"; [Net.ServicePointManager]::SecurityProtocol = [Net.ServicePointManager]::SecurityProtocol -bor 3072; Invoke-WebRequest -UseBasicParsing -Uri "https://aka.ms/azcmagent-wi
dows" -TimeoutSec 30 -OutFile "$env:TEMP\install_windows_azcmagent.ps1"; & "$env:TEMP\install_windows_azcmagent.ps1"; if ($LASTEXITCODE -ne 0) { exit 1; } & "$env:Program
432\AzureConnectedMachineAgent\azcmagent.exe" connect --resource-group "$env:RESOURCE_GROUP" --tenant-id "$env:TENANT_ID" --location "$env:LOCATION" --subscription-id "$env
:SUBSCRIPTION_ID" --cloud "$env:CLOUD" --correlation-id "$env:CORRELATION_ID"; } catch { $logBody = @({subscriptionId="$env:SUBSCRIPTION_ID";resourceGroup="$env:RESOURCE_G
UP";tenantId="$env:TENANT_ID";location="$env:LOCATION";correlationId="$env:CORRELATION_ID";authType="$env:AUTH_TYPE";operation="onboarding";messageType=$_.FullyQualifiedEr
orId;message="$_.Exception; }; Invoke-WebRequest -UseBasicParsing -Uri "https://gbl.his.arc.azure.com/log" -Method "PUT" -Body ($logBody | ConvertTo-Json) | out-null; Write-Host -Fo
egroundColor red $_.Exception; }
VERBOSE: Installing Azure Connected Machine Agent
VERBOSE: Total Physical Memory: 996 MB
VERBOSE: .NET Framework version: 4.8.4161
VERBOSE: Downloading agent package from https://aka.ms/AzureConnectedMachineAgent to
C:\Users\ADMINI~1\AppData\Local\Temp\2\AzureConnectedMachineAgent.msi
VERBOSE: Installing agent package
Installation of azcmagent completed successfully
INFO Connecting machine to Azure... This might take a few minutes.
INFO Please login using the pop-up browser to authenticate.

```

Figure 4.1: Installing azure arc agent on AWS instance.

- b. When prompted, provide the service principal values from step 2 and grant Azure Arc access to the AWS account.

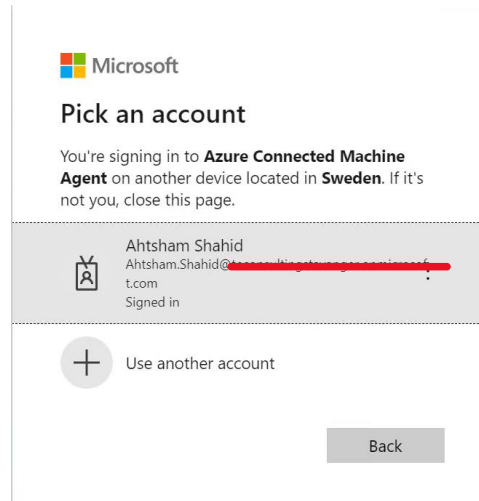


Figure 4.2 : Azure Portal credentials.

3. Verify that the AWS EC2 instance is connected to Azure Arc:
  - a. In the Azure portal, navigate to "Azure Arc" and select "Servers".
  - b. Verify that the AWS EC2 instance is listed under "Connected servers".

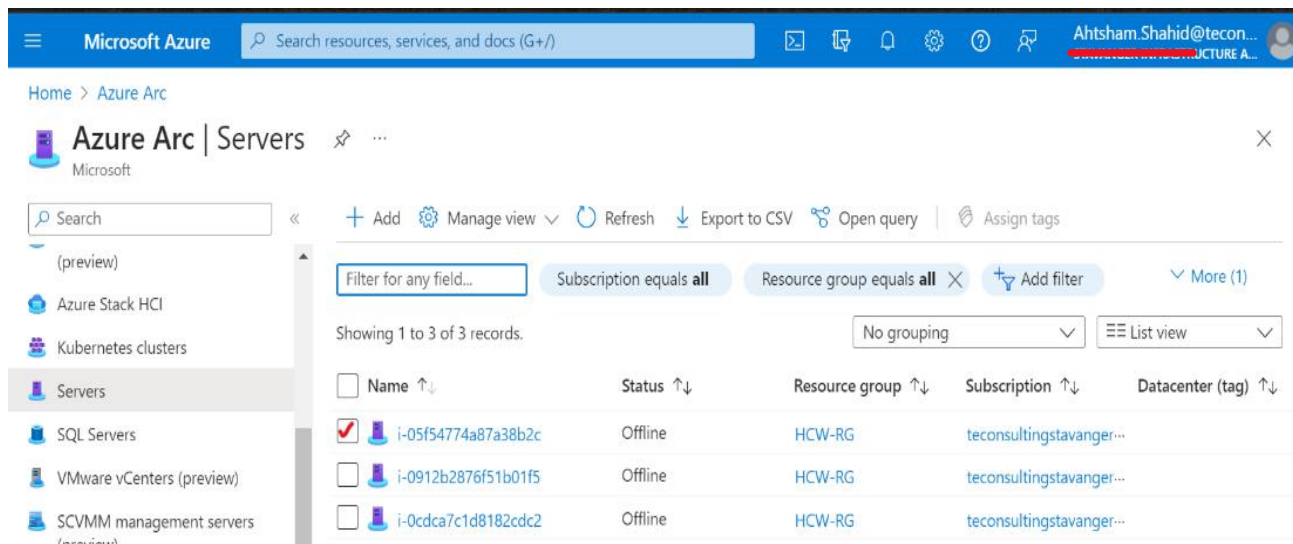


Figure 4.3: Listed AWS instances on Azure portal

The above lab has been implemented successfully and a virtual machine from AWS has been integrated to Azure Arc. It has enabled us to manage and monitor the hybrid environment resources from Azure portal. Reflections on the results and findings will be thoroughly discussed in chapter 5.

## 4.3 Azure Arc-enabled Kubernetes Clusters

This section involves creating a lab environment to onboard Kubernetes cluster from multiple platforms as Azure Arc-enabled k8s clusters.

For the following deployments, Ubuntu 20.04 has been used as the OS for the main server. Ubuntu is a popular, free, and widely used Linux distribution. It is stable, super secure and easy to use with a wide availability of online support[\[57\]](#).

### 4.3.1 On-prem Microk8s Cluster

The first scenario demonstrates onboarding of an existing running Kubernetes cluster running on on-premises.

Prerequisites:

1. One master node and at least one worker running on Windows/Linux machine.
2. Azure CLI: version 2.42.0 or above. If it is already installed, the cli version can be checked by running the `az --version` command.
3. A service principal (SP) in Azure with contributor role. The Azure SP is optional if you have the owner/contributor role at the Subscription level, but it is recommended to have a SP[\[58\]](#).
4. Azure resource providers:
  - a. "Microsoft.HybridCompute"
  - b. "Microsoft.GuestConfiguration"
  - c. "Microsoft.HybridConnectivity"

Deployment 1:

We need a master and 2 worker nodes to have Kubernetes on-prem setup. For this purpose, Multipass has been utilized to virtualize the Linux machine. Multipass is a tool used to get an instant Ubuntu VM [\[59\]](#). We have organized the lab in the following steps:

1. We create 3 nodes with multipass named as master, worker1 and worker2 using the following guide[\[60\]](#).

- We are using Microk8s which is a lightweight open-source Kubernetes distribution [61]. We installed microk8s on each of the nodes using the following commands.

For connecting to multipass nodes:

```
multipass shell <node>
```

Run on each node:

```
sudo snap install microk8s --classic
```

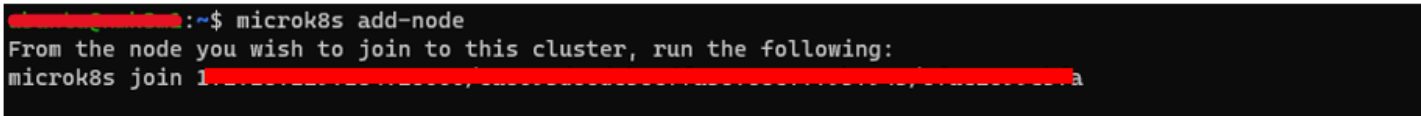
```
sudo usermod -aG microk8s $USER
```

```
sudo chown -f -R $USER ~/.kube
```

On master node only:

```
microk8s enable dns storage helm
```

```
microk8s add-node (run twice)
```



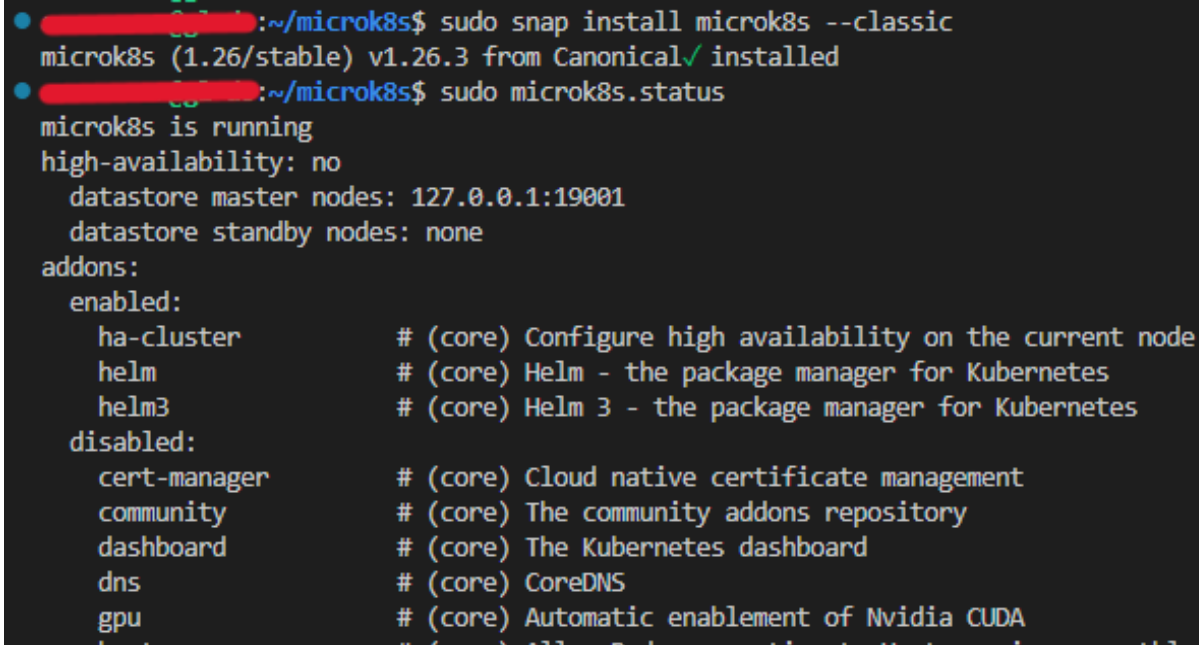
```

[redacted]@:~$ microk8s add-node
From the node you wish to join to this cluster, run the following:
microk8s join 1[redacted]a

```

Figure 4.4: Adding node with microk8s

Copy the outputs of the add-node command and run each output command on each worker node to join them with master node.



```

[redacted]@~/microk8s$ sudo snap install microk8s --classic
microk8s (1.26/stable) v1.26.3 from Canonical✓ installed
[redacted]@~/microk8s$ sudo microk8s.status
microk8s is running
high-availability: no
  datastore master nodes: 127.0.0.1:19001
  datastore standby nodes: none
addons:
  enabled:
    ha-cluster      # (core) Configure high availability on the current node
    helm            # (core) Helm - the package manager for Kubernetes
    helm3          # (core) Helm 3 - the package manager for Kubernetes
  disabled:
    cert-manager   # (core) Cloud native certificate management
    community      # (core) The community addons repository
    dashboard      # (core) The Kubernetes dashboard
    dns            # (core) CoreDNS
    gpu            # (core) Automatic enablement of Nvidia CUDA

```

Figure 4.5: microk8s installation

We can verify the joined nodes by entering the shell session of master node and running the command.





```

[REDACTED]:~$ subscriptionId=$(az account show --query id --output tsv)
[REDACTED]:~$ az ad sp create-for-rbac -n "azure-sp" --role "Contributor" --scopes /subscriptions/$subscriptionId
Creating 'Contributor' role assignment under scope '/subscriptions/[REDACTED]'
The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the credentials into your source control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "[REDACTED]",
  "displayName": "azure-sp",
  "password": "[REDACTED]",
  "tenant": "[REDACTED]"
}

```

Figure 4.8: creating a service principal

- In this step, we are going to register the required Azure providers by using the following commands and add the extensions connectedk8s and k8s-configuration. [62]

```

az provider register --namespace 'Microsoft.HybridCompute'
az provider register --namespace 'Microsoft.GuestConfiguration'
az provider register --namespace 'Microsoft.HybridConnectivity'
az extension add --name connectedk8s
az extension add --name k8s-configuration'

```

```

[REDACTED]:~$ az provider register --namespace Microsoft.Kubernetes
Registering is still on-going. You can monitor using 'az provider show -n Microsoft.Kubernetes'
[REDACTED]:~$ az provider register --namespace Microsoft.KubernetesConfiguration
Registering is still on-going. You can monitor using 'az provider show -n Microsoft.KubernetesConfiguration'
[REDACTED]:~$ az provider register --namespace Microsoft.ExtendedLocation
Registering is still on-going. You can monitor using 'az provider show -n Microsoft.ExtendedLocation'
[REDACTED]:~$ az provider show -n Microsoft.Kubernetes -o table
Namespace      RegistrationPolicy  RegistrationState
-----
Microsoft.Kubernetes  RegistrationRequired  Registered
[REDACTED]:~$ az provider show -n Microsoft.KubernetesConfiguration -o table
Namespace      RegistrationPolicy  RegistrationState
-----
Microsoft.KubernetesConfiguration  RegistrationRequired  Registered
[REDACTED]:~$ az provider show -n Microsoft.ExtendedLocation -o table
Namespace      RegistrationPolicy  RegistrationState
-----
Microsoft.ExtendedLocation  RegistrationRequired  Registered

```

Figure 4.99: Pre-requisite registration for Azure Arc

- Create a resource group on Azure using az cli in location westeurope. It can also be created via Azure portal.
- Deploy a sample application running nginx server on Kubernetes. The nginx deployment manifest[63] and figure how to deploy to the cluster can be found in Appendix A.4 [Deployment.yaml](#).

```

ubuntu@master:~$ kubectl apply -f - <<EOF
> apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2 # tells deployment to run 2 pods matching the template
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80

EOF
deployment.apps/nginx-deployment created
ubuntu@master:~$
ubuntu@master:~$ kubectl get all
NAME                                     READY   STATUS                    RESTARTS   AGE
pod/nginx-deployment-85996f8dbd-vs8mc   0/1     ContainerCreating        0           9s
pod/nginx-deployment-85996f8dbd-zzt8d   0/1     Pending                  0           9s

NAME                TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes  ClusterIP    10.152.183.1 <none>        443/TCP    36m

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment         0/2     2             0           11s

NAME                                     DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-deployment-85996f8dbd  2         2         0       11s
ubuntu@master:~$

```

Figure 4.1010: Sample deployment in K8s cluster

8. Export the environment variables stored in step 4 as shown below and login using service principal by running the following command.







The screenshot displays the Azure portal interface for a **microk8s-onprem** cluster. The top navigation bar shows the cluster name and a search bar. The left sidebar contains various management options like Overview, Activity log, and Settings. The main content area is divided into two sections: 'Workloads' and 'Namespaces'.

**Workloads Overview:**

| Name                      | Namespace   | Ready | Up-to-date | Available | Age        |
|---------------------------|-------------|-------|------------|-----------|------------|
| calico-kube-controllers   | kube-system | 1/1   | 1          | 1         | 49 minutes |
| coredns                   | kube-system | 1/1   | 1          | 1         | 47 minutes |
| hostpath-provisioner      | kube-system | 1/1   | 1          | 1         | 47 minutes |
| cluster-metadata-operator | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| clusterconnect-agent      | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| kube-aad-proxy            | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| metrics-agent             | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| clusteridentityoperator   | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| config-agent              | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| controller-manager        | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| extension-manager         | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| flux-logs-agent           | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| resource-sync-agent       | azure-arc   | 1/1   | 1          | 1         | 16 minutes |
| nginx-deployment          | default     | 2/2   | 2          | 2         | 13 minutes |

**Namespaces Overview:**

| Name              | Status | Age        |
|-------------------|--------|------------|
| kube-system       | Active | 49 minutes |
| kube-public       | Active | 49 minutes |
| kube-node-lease   | Active | 49 minutes |
| default           | Active | 49 minutes |
| azure-arc-release | Active | 18 minutes |
| azure-arc         | Active | 16 minutes |

Figure 4.17: Overview of connected on-prem cluster resources.

The above lab has been implemented successfully and Microk8s from on-premises has been integrated to Azure Arc. It has enabled us to manage and monitor the microk8s resources from Azure portal. Creating nodes with Multipass came with some performance challenges but were later resolved by resizing the ubuntu VM to a bigger size. Reflections on the results and findings will be thoroughly discussed in chapter 5.

In the next lab, we will integrate Azure Kubernetes Services (AKS) with Azure Arc.

### 4.3.2 Azure Kubernetes Service (AKS)

In this section, we are going to deploy Azure Kubernetes Service (AKS) and integrate it with Azure ARC. AKS is a managed Kubernetes service by Azure. It makes it easier to install a managed Kubernetes cluster on Azure and lowers the operational overhead and shifts it to Azure. Crucial tasks like health monitoring and maintenance are outsourced to Azure as a hosted Kubernetes service. A control plane is automatically deployed when you create an AKS cluster. Being a managed Azure resource that is hidden from the user, this control plane is offered free of charge. We are only responsible for the management of the nodes connected to the AKS cluster and for the cost associated with it[64].

Pre-requisite:

1. Azure CLI: version 2.42.0 or above. If it is already installed, the cli version can be checked by running the `az --version` command.
2. A Linux/windows machine. For this lab, we are using the same ubuntu machine used in the section [4.3.1]. (Optional: personal system can also be used)
3. Contributor access to an Azure subscription to create the AKS cluster.
4. A service principal (SP) in Azure with contributor role. (optional).
5. Azure resource providers registration:
  - a. Microsoft.HybridCompute
  - b. Microsoft.GuestConfiguration
  - c. Microsoft.HybridConnectivity

Deployment 2:

1. An AKS cluster can be created in multiple ways i.e., using terraform, ARM template and Azure Portal. As we do not require any highly advanced AKS cluster for this lab, creating an AKS cluster via Azure portal is a simple and straightforward process. By simply providing a few essential details and following the next steps, the cluster can be configured easily.

Home > Kubernetes services >

## Create Kubernetes cluster

Basics Node pools Access Networking Integrations Advanced Tags Review + create

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment, making it quick and easy to deploy and manage containerized applications without container orchestration expertise. It also eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand, without taking your applications offline. [Learn more](#)

**Project details**

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*

Resource group \*  [Create new](#)

**Cluster details**

Cluster preset configuration

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time. [Learn more and compare presets](#)

Kubernetes cluster name \*

Region \*

Availability zones

AKS pricing tier

Kubernetes version \*

Automatic upgrade

**Primary node pool**

The number and size of nodes in the primary node pool in your cluster. For production workloads, at least 3 nodes are recommended for resiliency. For development or test workloads, only one node is required. If you would like to add additional node pools or to see additional configuration options for this node pool, go to the 'Node pools' tab above. You will be able to add additional node pools after creating your cluster. [Learn more about node pools in Azure Kubernetes Service](#)

Node size \*

4 vcpus, 16 GiB memory  
Standard B4ms is recommended for dev/test configuration. [Change size](#)

Scale method \*  Manual  Autoscale

Autoscaling is recommended for dev/test configuration.

Node count range \*

[Review + create](#) [< Previous](#) [Next: Node pools >](#)

Figure 4.18: AKS creation Page

2. Login to Azure via CLI from ubuntu machine.
3. In this step, we try to speed up the deployment by creating a bash script which will register the required Azure providers, login with service principal created in Deployment 1 [section 4.3.1], update the kube config file (it is important when working with multiple clusters) and then connecting the cluster with Azure Arc. The bash script is given in Appendix [A.5 Bash Script: \[65\]](#)

Please note that we deployed in the same resource group as in Deployment 1 which is rg-Arc. The output of the script will look like below:







have control over the resources of the AKS cluster and can add/delete the resources from Azure Arc. Moreover, Azure Arc provides the monitoring of the cluster with different metrics. A more comprehensive analysis of the results obtained from this lab and potential capabilities of this integration will be detailed in chapter 5.

In the next section, we are going to explore Amazon Elastic Kubernetes Service and integrate it with Azure Arc.

### 4.3.3 Amazon Elastic Kubernetes Service (EKS)

In this section, we are going to deploy Amazon Elastic Kubernetes Service (EKS) and integrate it with Azure ARC. Amazon EKS is a managed k8s service to run Kubernetes clusters in the AWS cloud. The Kubernetes control plane nodes in the cloud oversee the features like scaling out/in of containers, assuring application availability, cluster storage and networking. These nodes are automatically managed by Amazon EKS for availability and scalability. With integrated tooling and straightforward deployment to Amazon Outposts, or virtual machines, EKS offers a consistent, fully supported Kubernetes solution on-premises.[\[66\]](#)

#### Pre-requisites

1. A Linux/windows machine. For this lab, we are using the same ubuntu machine used in previous deployments.
2. An IAM role with sufficient roles/policies for EKS as well as EC2.
3. Aws CLI version 2.
4. A service principal (SP) in Azure with contributor role. (optional).
5. Azure resource providers registration:
  - a. Microsoft.HybridCompute
  - b. Microsoft.GuestConfiguration
  - c. Microsoft.HybridConnectivity

#### Deployment 3

1. Install AWS CLI using the following commands[\[67\]](#).

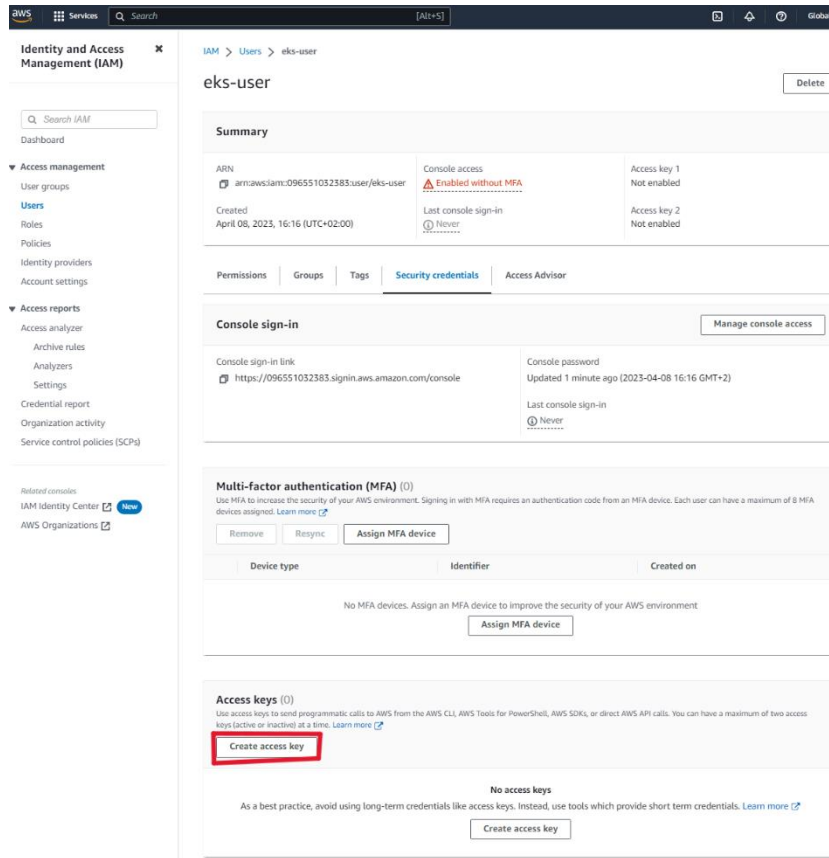
```
1) curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
2) unzip awscliv2.zip
3) sudo ./aws/install
```

2. Create a user named eks-user in IAM via AWS portal like below:

The image shows two screenshots from the AWS IAM console. The top screenshot displays the 'Users' page, which is currently empty, showing a search bar and a table with columns for User name, Groups, Last activity, MFA, Password age, and Active key age. The bottom screenshot shows the 'Specify user details' form for creating a new user. The 'User name' field is filled with 'eks-user'. There is a checkbox for 'Provide user access to the AWS Management Console - optional' which is unchecked. A blue callout box contains the text: 'If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more'. The 'Next' button is highlighted in orange.

Figure 4.22: User creation for EKS

3. Login to aws console again as eks-user and create the access for eks-user to authenticate the ubuntu server to use aws-cli. Follow the screenshots below.



IAM > Users > eks-user > Create access key

Step 1  
**Access key best practices & alternatives**

Step 2 - optional  
 Set description tag

Step 3  
 Retrieve access keys

### Access key best practices & alternatives

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

- Command Line Interface (CLI)**  
 You plan to use this access key to enable the AWS CLI to access your AWS account.
- Local code**  
 You plan to use this access key to enable application code in a local development environment to access your AWS account.
- Application running on an AWS compute service**  
 You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Figure 4.23: Creating aws-cli access for ubuntu server

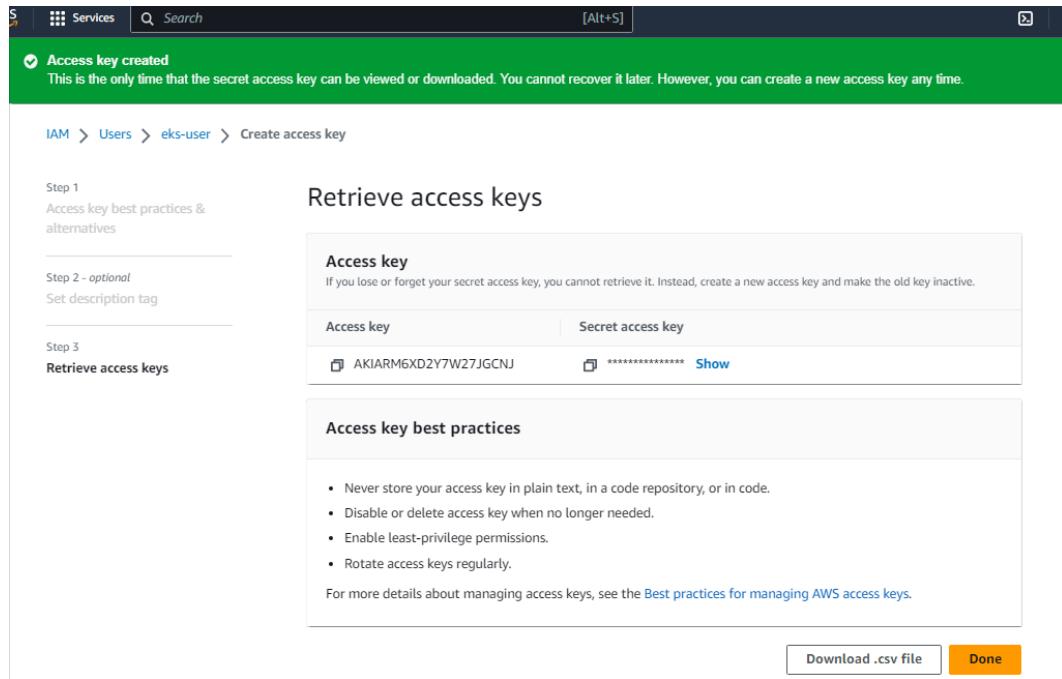


Figure 4.24: Access key

4. Store the Access Key and Secret Access Key.
5. Login to AWS portal via AWS CLI by running `aws configure`.

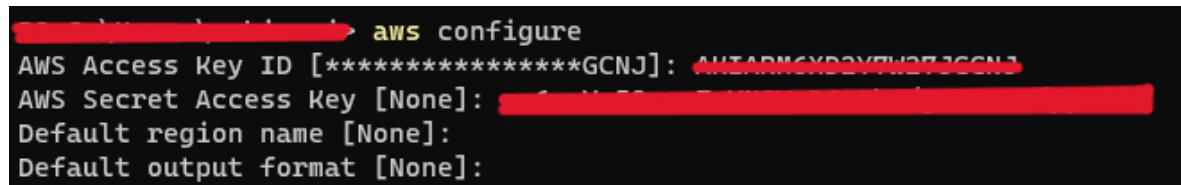


Figure 4.25: Logging in to aws environment with aws-cli

6. Now back to AWS portal, grant eks-user with the following permissions from IAM. This is important for creating the EKS cluster and nodes in that cluster.

The screenshot shows the AWS IAM console interface for the user 'eks-user'. The left sidebar contains navigation options for Identity and Access Management (IAM), including Access management, Access reports, and Related consoles. The main content area displays the user's summary, console access status, and a list of permissions policies. The 'Permissions policies' section is expanded, showing a table of six AWS managed policies attached to the user.

| Policy name                        | Type                       | Attached via        |
|------------------------------------|----------------------------|---------------------|
| AdministratorAccess                | AWS managed - job function | Directly            |
| AmazonEC2ContainerRegistryReadOnly | AWS managed                | Group eks           |
| AmazonEKS_CNI_Policy               | AWS managed                | Group eks           |
| AmazonEKSClusterPolicy             | AWS managed                | Directly, Group eks |
| AmazonEKSServicePolicy             | AWS managed                | Group eks           |
| AmazonEKSWorkerNodePolicy          | AWS managed                | Group eks           |

Figure 4.26: Assigning IAM roles to eks-user

7. Creating an EKS cluster via AWS portal is a simple and straightforward process. By simply providing a few essential details and following the next steps, the cluster can be configured easily.

The screenshot shows the AWS Management Console interface for creating an EKS cluster. The breadcrumb navigation is 'EKS > Clusters > Create EKS cluster'. The left sidebar shows a six-step process: Step 1 (Configure cluster), Step 2 (Specify networking), Step 3 (Configure logging), Step 4 (Select add-ons), Step 5 (Configure selected add-ons settings), and Step 6 (Review and create). The main content area is titled 'Configure cluster' and contains three sections: 'Cluster configuration', 'Secrets encryption', and 'Tags (0)'. The 'Cluster configuration' section includes a text input for the cluster name (set to 'eks-demo'), a dropdown for the Kubernetes version (set to '1.25'), and a dropdown for the cluster service role (set to 'myAmazonEKSClusterRole'). The 'Secrets encryption' section has a radio button for 'Turn on envelope encryption of Kubernetes secrets using KMS', which is currently unselected. The 'Tags' section indicates that no tags are currently added and provides an 'Add tag' button. At the bottom right, there are 'Cancel' and 'Next' buttons.

Figure 4.27: EKS creation page

8. Unlike AKS, EKS installation does not automatically deploy any nodes/resources i.e., no workload and it is not possible to just integrate an empty cluster to Azure Arc.
9. So, we create a node group to populate the cluster as shown below. It will create 2 EC2 instances and a few other cluster resources associated with the node.



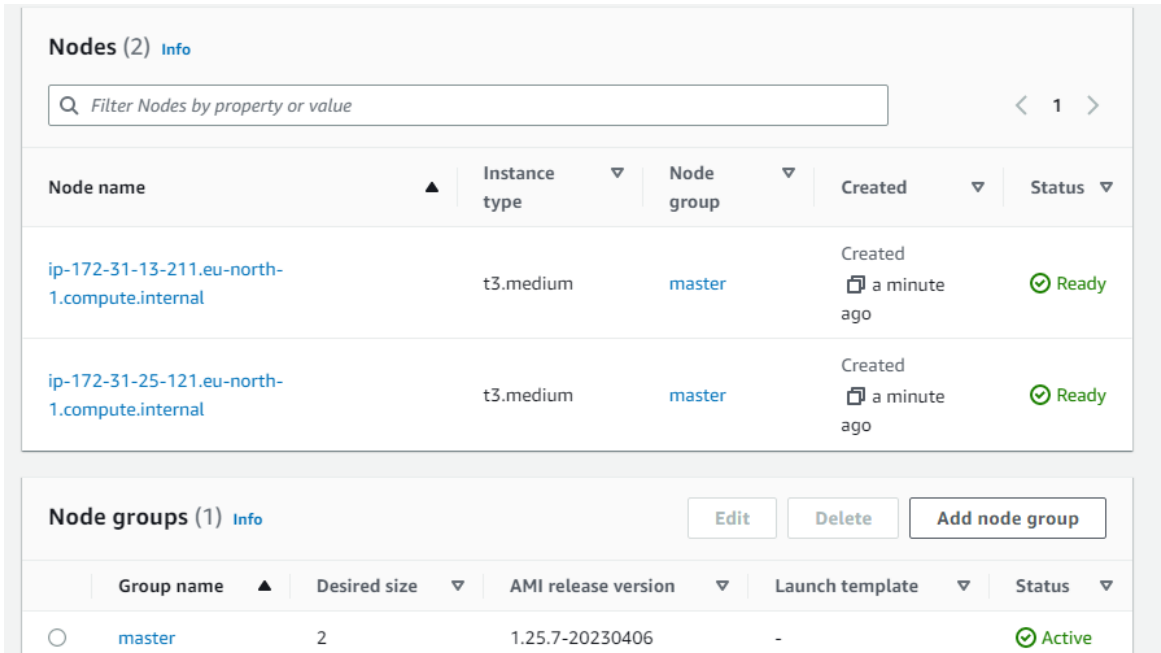


Figure 4.28: Node group

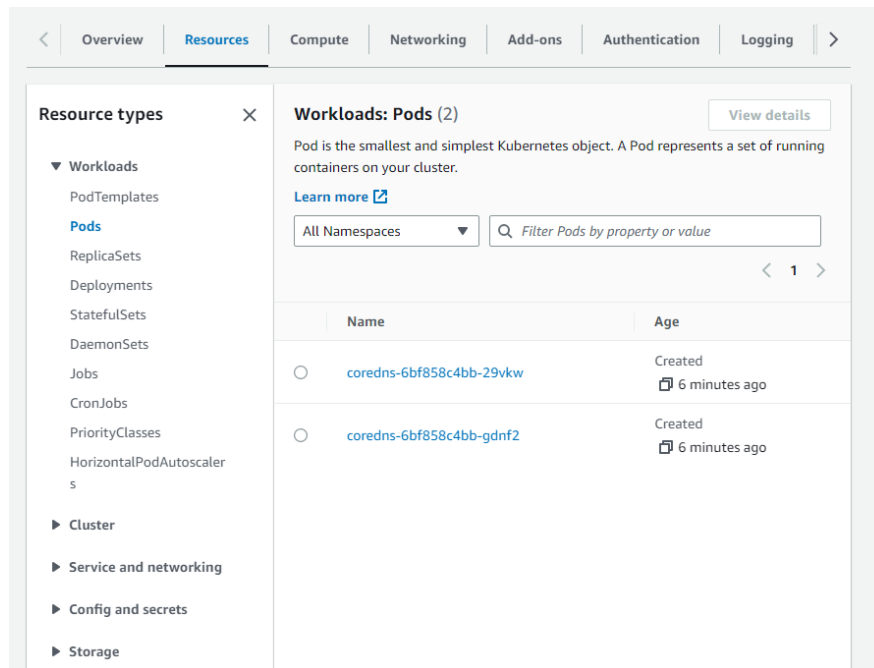


Figure 4.29: Workload on EKS-Node

- From ubuntu machine, we import configuration details of this cluster using aws cli tool like we did in previous 2 deployments.

```

azureuser@gl-do:~$ aws eks update-kubeconfig --region eu-north-1 --name eks-demo
Updated context arn:aws:eks:eu-north-1:096551032383:cluster/eks-demo in /home/azureuser/.kube/config
azureuser@gl-do:~$ kubectl
aks-demo
arn:aws:eks:eu-north-1:096551032383:cluster/eks-demo
azureuser@gl-do:~$ kubectl get nodes
NAME                                STATUS    ROLES    AGE     VERSION
ip-172-31-13-211.eu-north-1.compute.internal Ready    <none>   6m10s   v1.25.7-eks-a59e1f0
ip-172-31-25-121.eu-north-1.compute.internal Ready    <none>   6m12s   v1.25.7-eks-a59e1f0
azureuser@gl-do:~$ kubectl get pods --namespace kube-system
NAME                                READY    STATUS    RESTARTS   AGE
aws-node-5gkrj                      1/1     Running   0           2m31s
aws-node-75w9w                      1/1     Running   0           2m35s
coredns-6bf858c4bb-29vkw            1/1     Running   0           12m
coredns-6bf858c4bb-gdnf2            1/1     Running   0           12m
kube-proxy-4z6xk                    1/1     Running   0           6m18s
kube-proxy-pj6xs                    1/1     Running   0           6m20s

```

Figure 4.30: Overview of EKS Cluster on Ubuntu

11. Repeat step 5 in Deployment 1 to register Azure providers and then login using az cli with the same service principle we created before. We do not have to create a resource group and add connectk8s extensions again as we are using the same environment on Azure.
12. Connect EKS cluster with Azure Arc by running the following command as we did in previous deployments.

```
az connectedk8s connect --name "EKS-AWS" --resource-group "rg-Arc" --location "westeurope" --tags "EKS"
```

```

azureuser@gl-do:~$ az connectedk8s connect --name "EKS-AWS" --resource-group "rg-Arc" --location "westeurope" --tags "EKS"
This operation might take a while...

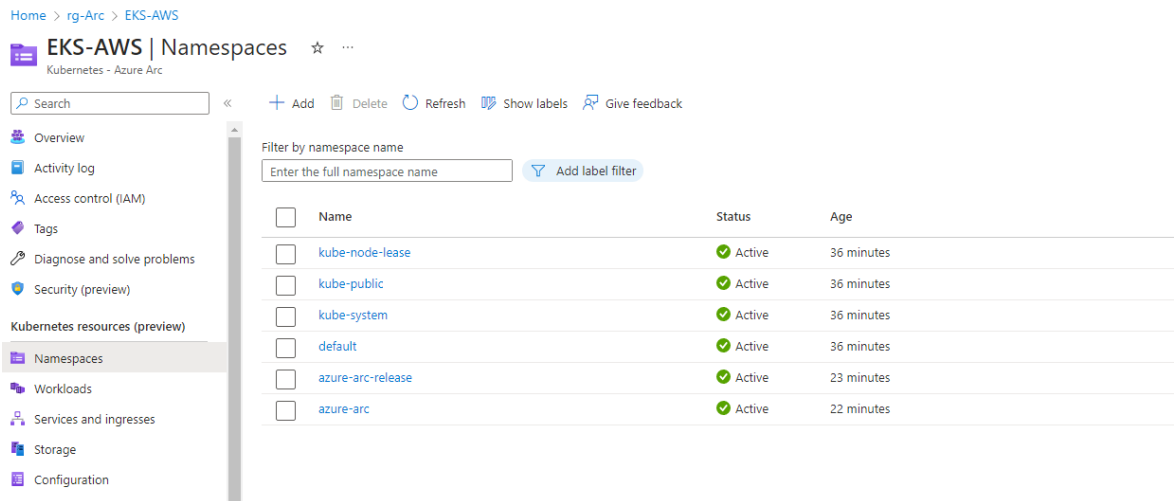
Unable to fetch the Object ID of the Azure AD application used by Azure Arc service. Unable to enable the 'custom-locations' feature. Insufficient privileges to complete the operation.
{
  "agentPublicKeyCertificate": "MIICGgKAgEAtzuga3myICDPLYhtyJef8SGDtAz0ILsPyCrZ0Spg8Qug+ZkYU+HJMCCQY9sng5MgvHnHM1HpIekM0Noappiq0pkzBXVDiwLo68JTZTZh2UB1C0KGm7T08uEXBR2jug8JAiRvSSqD09UjAMEciJT6Jd7A1XAjSS13czevLqeJ2UTDctHn1eN78/UEwcdBuhrgsNvvGLei00wexOz9PIK8Hj7wDJR2sTzBdtZ7E7EAbPtIlg+kdOEPPLyPHdvSEvVTXmBdVXRsfCUImljs07o8Gm6ZILNkIGuPJ+FYjNIsPukV4/TAXATSAHssZkIna0JUzJGJE0V7HDH+ZHyAbP675UjpkRqB6yq4vt+N5gsey9u0gnxZUZU09Pp+zaz+zdssLccVxRPYSHLFcmRWS68mzHqylws5zZd19HcuplplusLGS1b7eLHZc0pjjyo/UMt0Yr85XuHKVRd3uKAHbmdLtfWpPrzPaIhXIXNbo1PwwWmdmW5KHqZ0zObH6L08ggn+TUoAk4sNwqdtEk1Y3zFoDLG0pn15AHc1mvV1QeJGhhq7KzBeGHqInk+DnSB+hyhPy51LqaqV2IDmK6mzQ5jkC0dwmFm8buKCN6JCLh8gzUvUIdQkWSRgP8tUemLkEYvgexK055ArB+KjztqC/buVcbpfRkG9eGR3:kCAwEAAQ==",
  "agentVersion": null,
  "connectivityStatus": "Connecting",
  "distribution": "eks",
  "id": "/subscriptions/27af7ba7-753a-4422-8296-e16a15e0b471/resourceGroups/rg-Arc/providers/Microsoft.Kubernetes/connectedClusters/EKS-AWS",
  "identity": {
    "principalId": "d6e9872f-2660-44de-917f-f2cbec79c9f1",
    "tenantId": "49517e6b-4fea-4df5-ba75-84dabal539b9",
    "type": "SystemAssigned"
  },
  "infrastructure": "aws",
  "kubernetesVersion": null,
  "lastConnectivityTime": null,
  "location": "westeurope",
  "managedIdentityCertificateExpirationTime": null,
  "name": "EKS-AWS",
  "offering": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "rg-Arc",
  "systemData": {
    "createdAt": "2023-04-08T16:24:26.660074+00:00",
    "createdBy": "47834fac-d536-45a2-bf1a-e74a725a5ddf",
    "createdByType": "Application",
    "lastModifiedAt": "2023-04-08T16:24:26.660074+00:00",
    "lastModifiedBy": "47834fac-d536-45a2-bf1a-e74a725a5ddf",
    "lastModifiedByType": "Application"
  },
  "tags": {
    "EKS": ""
  },
  "totalCoreCount": null,
  "totalNodeCount": null,
  "type": "microsoft.kubernetes/connectedclusters"
}
azureuser@gl-do:~$

```

Figure 4.31: Using Connectedk8s to onboard EKS cluster.

13. Repeat steps 11-13 from Deployment 1 to authenticate Azure Arc to see the resources of EKS cluster.

14. Integration of EKS with Azure Arc can be seen below.

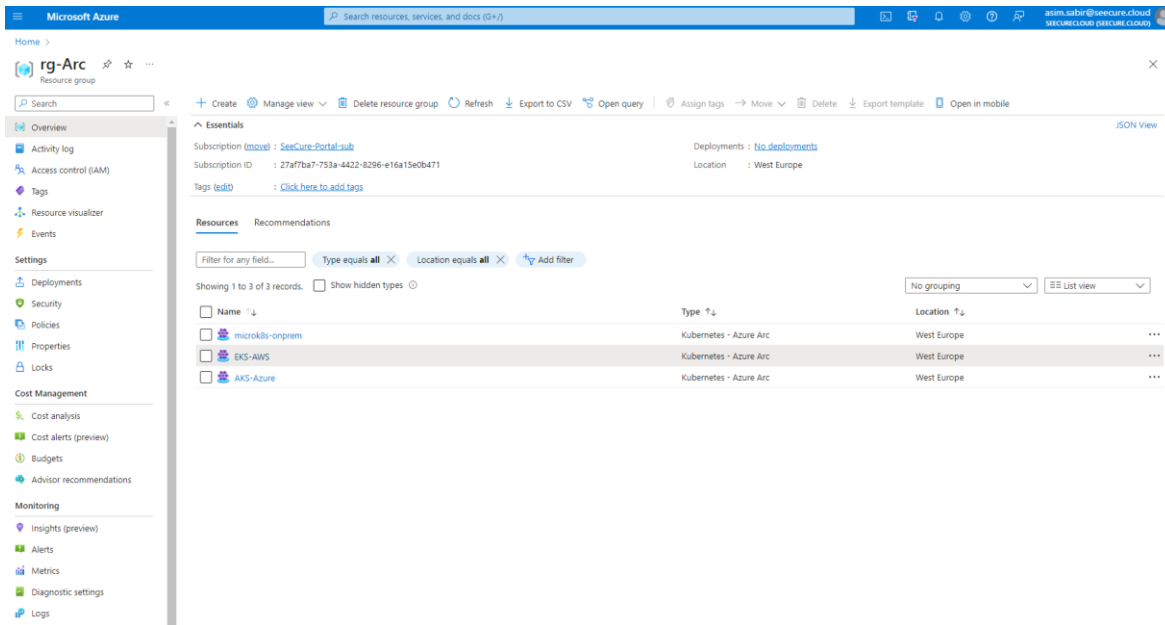


The screenshot shows the Azure portal interface for the 'EKS-AWS | Namespaces' resource group. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security (preview), and Kubernetes resources (preview). The main content area displays a table of namespaces with the following data:

| Name              | Status | Age        |
|-------------------|--------|------------|
| kube-node-lease   | Active | 36 minutes |
| kube-public       | Active | 36 minutes |
| kube-system       | Active | 36 minutes |
| default           | Active | 36 minutes |
| azure-arc-release | Active | 23 minutes |
| azure-arc         | Active | 22 minutes |

Figure 4.32: Overview of connected EKS cluster resources

We have now successfully connected On-Prem Microk8s, AKS and EKS with Azure Arc.



The screenshot shows the Azure portal interface for the 'rg-Arc' resource group. The 'Resources' section displays a table of resources with the following data:

| Name            | Type                   | Location    |
|-----------------|------------------------|-------------|
| microk8s-onprem | Kubernetes - Azure Arc | West Europe |
| EKS-AWS         | Kubernetes - Azure Arc | West Europe |
| AKS-Azure       | Kubernetes - Azure Arc | West Europe |

Figure 4.33: On-Prem, AKS and EKS integrated with Azure Arc

Connecting the EKS cluster with Azure Arc is comparatively complex due to the required permission, IAM roles and the fact that the node needs to be deployed manually. After the integration, we can oversee the resources of the EKS cluster and can update the cluster from Azure Arc. Detailed insights of the outcomes of this deployment and the integration's prospective capabilities and features will be discussed in Chapter 5.

## 4.4 Deployment of Arc-Enabled SQL Server

This section will walk you through deployment of Microsoft SQL Server as an Azure Arc-Enabled SQL Server. We plan to implement this lab with the help of Terraform which helps in automation and complex deployment of the infrastructure on cloud. Terraform is also known as Infrastructure as a code service. In this lab an AWS EC2 instance with SQL Server installed is deployed with the help of [68] terraform:

Pre-Requisites:

First, we identified the terraform version being used on cloud shell.

Terraform `--version`.

Version was outdated, to update the version we followed the following steps below.

1. Copy the link for AMD64 version from HASHICORP website.
2. Run the curl command (`curl -O <terraform_download_url>`)
3. Unzip the file (`unzip <zip_file_downloaded_in_previous_step>`)
4. New directory (`mkdir bin`)
5. Move the terraform file into the bin directory (`mv terraform bin/`)
6. Close and restart the cloud shell.

```
PS /home/ahtsham> terraform version
Terraform v1.3.2
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.4.4. You can update by downloading from https://www.terraform.io/downloads.html
PS /home/ahtsham> curl -O https://releases.hashicorp.com/terraform/1.4.4/terraform_1.4.4_darwin_amd64.zip
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 21.0M  100 21.0M    0     0  111M    0 --:--:-- --:--:-- --:--:--  112M
PS /home/ahtsham> unzip
terraform_1.4.4_darwin_amd64.zip  Microsoft/          DigiCertGlobalRootCA.crt.pem  terraform_1.4.4_darwin_amd64.zip
clouddrive/                    clouddrive/
PS /home/ahtsham> unzip terraform_1.4.4_darwin_amd64.zip
Archive:  terraform_1.4.4_darwin_amd64.zip
  inflating: terraform
PS /home/ahtsham> mkdir bin
PS /home/ahtsham> mv terraform bin/
```

Figure 4.34: Update the terraform version.

7. After the terraform installation, we need to register the resource providers below.
  - a. `az provider register --namespace Microsoft.AzureArcData`

b. `az provider register --namespace Microsoft.HybridCompute`

8. Registration process can be monitored with the following commands below:

a. `az provider show -n Microsoft.AzureArcData -o table`

b. `az provider show -n Microsoft.HybridCompute -o table`

```
PS /home/ahtsham> az provider register --namespace Microsoft.AzureArcData
PS /home/ahtsham> az provider register --namespace Microsoft.HybridCompute
PS /home/ahtsham> az provider show -n Microsoft.AzureArcData -o table
Namespace                RegistrationPolicy        RegistrationState
-----
Microsoft.AzureArcData    RegistrationRequired      Registered
PS /home/ahtsham>
```

Figure 4.35: Register the resource providers.

9. New AWS IAM Role & Key:

- a. In the IAM console, click on "Users" in the left-hand menu.
- b. Find the user associated with the IAM role you just created and click on their name.
- c. Click on the "Security credentials" tab.
- d. Scroll down to the "Access keys" section and click "Create access key".
- e. Save the access key ID and secret access key to a safe location. You will not be able to view the secret access key again after you leave this page.

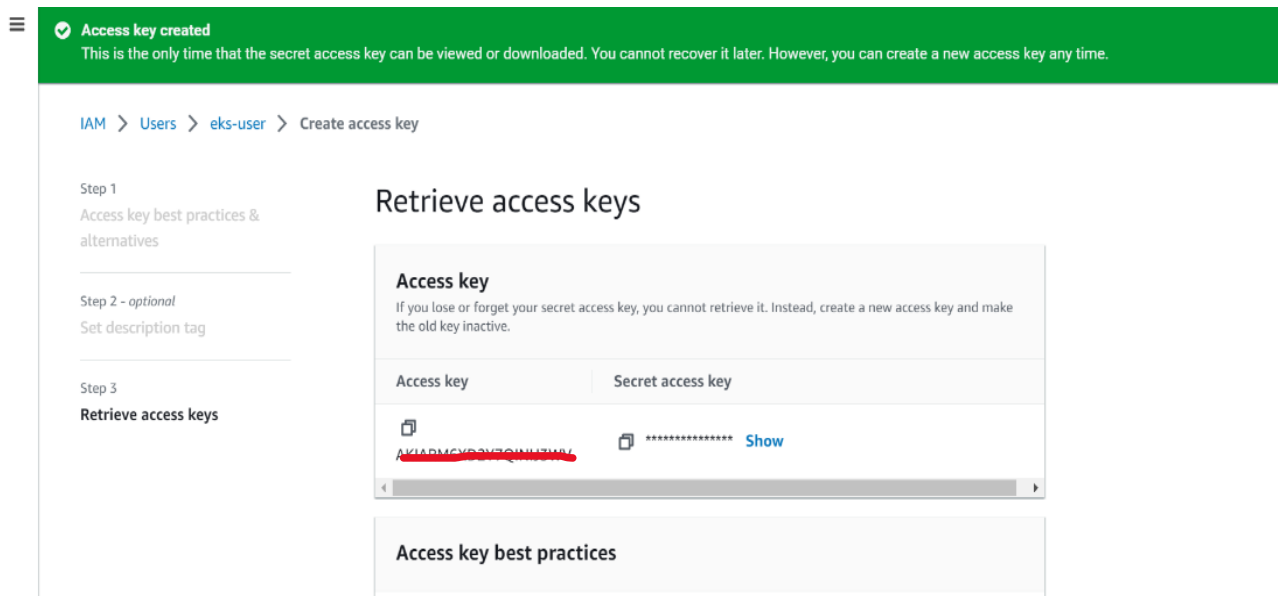


Figure 4.36: Access key creation.

Deployment:

AWS EC2 Microsoft Server instance with SQL Server and Terraform is being used to connect it to the Azure Arc. In the end of this session, we will have an SQL Server installed on AWS EC2 instance and connected to our Azure portal for different kind of assessments.

To deploy an EC2 instance with SQL Server and onboard it as an Azure Arc-enabled SQL Server using Terraform, you can follow these steps:

1. Create an IAM role for the EC2 instance that has the necessary permissions to interact with the AWS resources and Azure Arc-enabled servers.
2. Created a Terraform configuration file with the following resources:
3. AWS EC2 instance resource to launch an instance with SQL Server installed.
4. Provisioner resource to execute a shell script on the EC2 instance to install and configure the Azure Arc agent.
5. Configure the Terraform provider for Azure Arc to connect to your Azure Arc-enabled SQL Server instance.
6. Run the Terraform apply command to deploy the EC2 instance and onboard it as an Azure Arc-enabled SQL Server.

Terraform code used to deploy the resources and services can be found Appendix [A.2](#)

[Terraform Code](#). [\[69\]](#)

Below command helps to deploy the resources via terraform:

- a. `terraform init`
- b. `terraform apply -auto-approve`

After the instance deployment the following script has been used from the GitHub repository mentioned to install the SQL Server[\[70\]](#).

Script Name: sql.ps1

GitHub Repo: [GitHub - repo](#)

Run SQL Script:

```

The install of azure-cli was successful.
Software installed as 'msi', install location is likely default.

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
Installing sql-server-management-studio
Chocolatey v1.3.1
Installing the following packages:
sql-server-management-studio
By installing, you accept licenses for the packages.
Progress: Downloading sql-server-management-studio 19.0.20209.0... 100%
sql-server-management-studio v19.0.20209.0 [Approved]
sql-server-management-studio package files install completed. Performing other installation steps.
Downloading sql-server-management-studio
from 'https://download.microsoft.com/download/9/f/8/9f8197f4-0f71-42a3-8717-b2817c77b820/ssms-Setup-ENU.exe'
Download of SSMS-Setup-ENU.exe (628.83 MB) completed.
Hashes match.
Installing sql-server-management-studio...

```

Figure 4.37: SQL Server Management Studio Installation.

SQL Successfully installed:

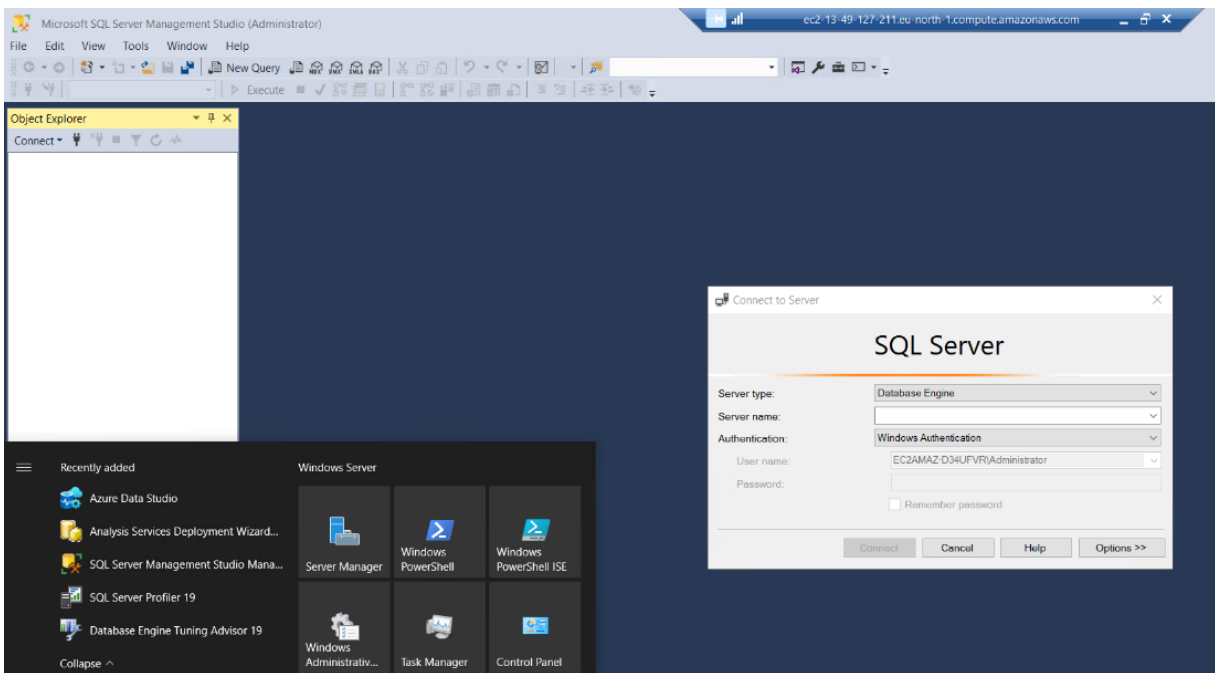


Figure 4.38: SQL Server Management Studio.

Onboarding the SQL Server to Azure Portal: See Appendix [A.3 SQL installation Script](#). [71]

The following script is copied from the Azure Arc portal to onboard the sql server to the portal and to install the agent on the AWS machine.

In the picture below, we are running the script copied from the Azure Arc module to connect Azure Arc-enabled SQL Server.

```

Administrator: Windows PowerShell
>> & "$env:ProgramW6432\AzureExtensionForSQLServer\AzureExtensionForSQLServer.exe" --subId $subId --resourceGroup $resourceGroup --location $location --tenantid $servicePrincipalTenantId --service-principal-app-id $servicePrincipalAppId --service-principal-secret $servicePrincipalSecret --proxy $proxy --licenseType $licenseType
>> } else {
>> & "$env:ProgramW6432\AzureExtensionForSQLServer\AzureExtensionForSQLServer.exe" --subId $subId --resourceGroup $resourceGroup --location $location --tenantid $tenantId --proxy $proxy --licenseType $licenseType
>> }
>> }
>> if($LASTEXITCODE -eq 0){
>> Write-Host -ForegroundColor green "Azure extension for SQL Server is successfully installed. If one or more SQL Server instances are up and running on the server, Arc-enabled SQL Server instance resource(s) will be visible within a minute on the portal. Newly installed instances or instances started now will show within an hour."
>> }
>> else{
>> $message = "Failed to install Azure extension for SQL Server. Please see $currentDir\AzureExtensionForSQLServerInstallation.log file for more information."
>> Write-Host -ForegroundColor red $message
>> }
>> }
>> catch {
>> Write-Host -ForegroundColor red $_.Exception
>> throw
>> }
To sign in, use a web browser to open the page https://microsoft.com/device/login and enter the code C5DMR5F4L to authenticate.
Authentication successful.
Onboarding SQL Server to Azure Arc. It will take few minutes. Please see C:\Users\Administrator\AzureExtensionForSQLServerInstallation.log file for more information.
Installing Azure Connected Machine Agent.

```

Figure 4.39: Script to connect Arc-enabled SQL to Azure Arc.

Once the SQL Server is onboarded to the azure arc module, the picture below shows the new onboarded machine and extensions in portal.

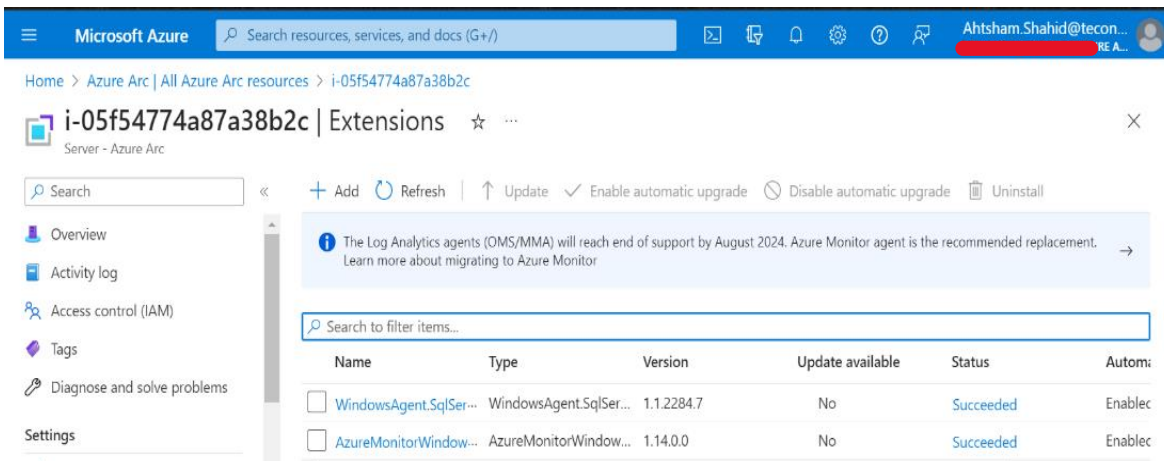


Figure 4.40: SQL Machine and extensions.

To conclude, connecting the SQL Server machine requires multiple steps, especially if we are onboarding with the help of Terraform. This lab is comparatively complex due to the required permission, IAM roles and writing the terraform code. After the integration, we can oversee the resources of the SQL Server in Azure Arc. Detailed insights of the outcomes of this deployment and the integration's prospective capabilities and features will be discussed in Chapter 5.



# CHAPTER 5

## 5 Results and Discussions

In this chapter we will write about our results and findings and will discuss them in detail. As we had a detailed overview about Azure Arc in previous chapters, which is a service provided by Microsoft Azure that allows customers to extend Azure services and management to any infrastructure.

### 5.1 Introduction

We can simply divide our findings into four different areas.

**Simplified management:** Azure Arc provides a centralized management portal for on-premises, multi-cloud, and edge environments. This simplifies management and reduces the need for multiple tools and consoles. Also, it gives a great opportunity for automation. One can implement policies, monitoring services and do the updating on multiple virtual machines shown in below figure.

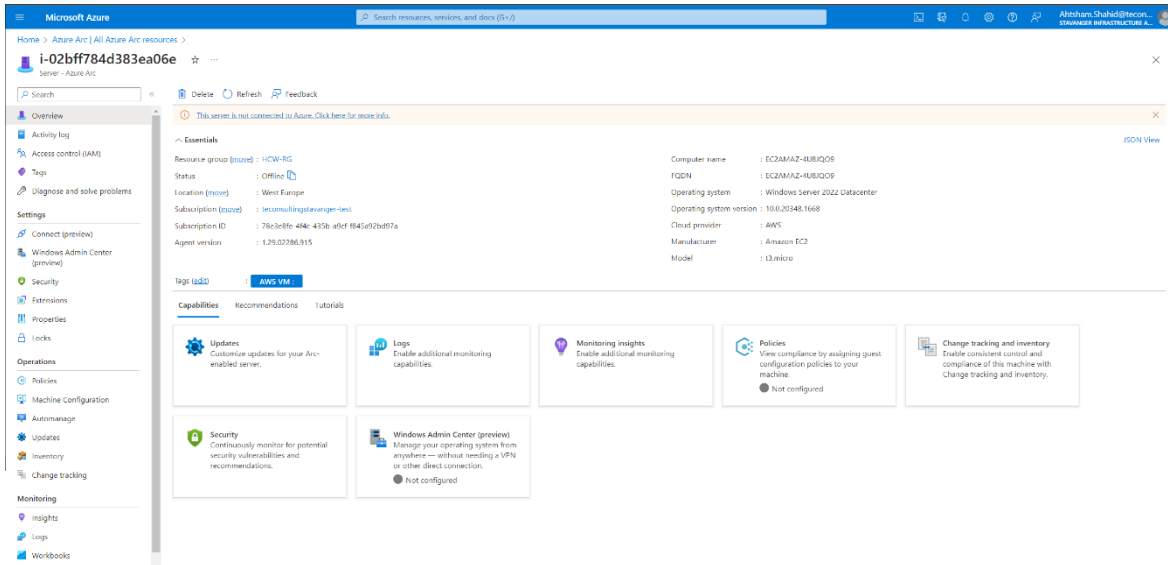


Figure 5.1: Azure Arc capabilities.

**Hybrid cloud readiness:** Azure Arc enables customers to run Azure services on any infrastructure, including on-premises servers and Kubernetes clusters, as well as public clouds like AWS and Google Cloud. This makes it easier for customers to move workloads to and from the cloud as needed. As we onboarded multiple virtual machines from different environments on our azure portal in chapter 4.

**Improved security:** Azure Arc provides security features such as Azure Policy, Azure Security Center, and Azure Defender to protect workloads running on any infrastructure. This helps ensure that customers can comply with regulatory requirements and maintain the security of their workloads. As our scope was to onboard the machines from multiple environments suggested by our industrial supervisor to have a look and explore the azure arc service.

**Streamlined deployment:** Azure Arc provides a streamlined deployment process that allows customers to deploy Azure services on any infrastructure with a few clicks. This makes it easier for customers to adopt new Azure services and features. Either it's a single server or bunch of servers together one only needs to download onboard script from azure arc and run it into the servers.

## 5.2 Policies with Azure Arc

Azure Arc introduces a powerful feature that enables users to apply policies across diverse environments, promoting governance, compliance, and resource efficiency. These policies, created using Azure Policy, offer a means to enforce best practices, security standards, and regulatory requirements. Expressed in JSON format, Azure Arc policies can be assigned to various entities, including resource groups, subscriptions, management groups, and individual resources.

In our previous chapter, while we successfully deployed Azure Arc-enabled resources and explored their features, we did not delve into the enforcement of Azure policies on these resources. In this chapter, we will delve into the immense potential and capabilities of Azure Arc in terms of policy enforcement for Azure Arc-enabled resources. Azure Arc empowers users to effectively manage and enforce policies on their Azure Arc-enabled resources. Azure policies act as rule sets that dictate the configuration and behavior of Azure resources. With Azure Arc, these policies can be extended to ensure compliance, security, and governance throughout the hybrid environment. Regardless of their location, be it on-premises, multi-cloud, or edge environments, Azure Arc enables the definition and enforcement of Azure policies on diverse resources[6]. A notable advantage of leveraging Azure policies with Azure Arc-enabled resources is the central management and enforcement of policies across hybrid environments. This ensures consistent compliance with organizational policies and regulations, regardless of the resource's location. Azure Arc further simplifies policy management by providing a unified policy enforcement mechanism that seamlessly operates across multiple environments, alleviating complexities for IT teams [72].

The application of Azure policies in conjunction with Azure Arc also enhances security practices. These policies can enforce measures such as robust password and access control configurations, network security group implementations for traffic restrictions, and activation of security features like encryption and threat detection. Moreover, enforcing Azure policies on Azure Arc-enabled resources aids in meeting regulatory requirements. Numerous regulations necessitate specific security controls and governance policies. Azure policies can effectively enforce these controls throughout hybrid environments, ensuring compliance with regulatory obligations.

To enforce policies on Azure Arc-enabled resources, policies must be created within Azure Policy and assigned to the corresponding resources. These assignments can be done at the individual resource level, for resource groups, or even entire subscriptions. Azure Policy continuously evaluates resources against assigned policies to maintain compliance. Non-compliant resources are flagged and can be remediated either automatically or manually by IT teams. Although our lab environment did not enforce any policies on Azure Arc-enabled resources, we can confidently conclude, based on Azure Arc's capabilities, that Azure policies can be seamlessly enforced on these resources regardless of their location. Azure Arc offers a consistent policy enforcement mechanism that functions harmoniously across diverse environments, providing IT teams with simplified policy management.

### 5.3 Security & Governance

In this discussion, we will explore the potential of Azure Arc in terms of security and governance, with a specific focus on two prominent features: Azure Active Directory (AAD) and Centralized Access Control. These features play a crucial role in establishing robust security measures and streamlined governance practices within the Azure Arc ecosystem.

**Azure Active Directory (AAD):** AAD serves as a cloud-based identity and access management service, offering seamless single sign-on and multi-factor authentication capabilities for a wide range of cloud-based applications and services. With Azure Arc, AAD can be utilized to manage identity and access for resources managed by Azure Arc, regardless of their geographical location. By harnessing the power of AAD, users can establish role-based access control (RBAC) policies for Azure Arc-enabled resources, granting finely grained access privileges to different stakeholders. Additionally, AAD facilitates the enforcement of robust password policies, bolstering the security of user access credentials[12].

Although our lab environment did not involve the configuration of AAD integration with Azure Arc, it is worth noting that in real-world scenarios, AAD integration with Azure Arc offers a secure and centralized identity management solution across hybrid cloud environments. Through

AAD, organizations can ensure that only authorized personnel possess the necessary access to specific resources and applications, thereby reducing the potential attack surface for security threats.

Centralized access control: It is another noteworthy security and governance feature provided by Azure Arc. This feature empowers users to define access policies for their resources, which can be effectively enforced across all environments managed by Azure Arc.

Azure Arc's centralized access control mechanism operates by creating comprehensive policies that govern who can access specific resources, what actions they can perform, and when they can access them. These policies are uniformly enforced across all Azure Arc-managed environments, ensuring consistent security practices and governance measures throughout hybrid cloud environments. While our lab setup did not involve the configuration of centralized access control policies, it is important to highlight that in real-world scenarios, leveraging centralized access control within Azure Arc can ensure the implementation of uniform security policies across all resources within a hybrid cloud environment, thereby minimizing the potential attack surface. Overall, Azure Arc's integration with Azure Active Directory and its provision of centralized access control present powerful tools for establishing robust security measures and efficient governance practices within hybrid cloud environments.

## 5.4 Automation

Automation plays a pivotal role in cloud management, and Azure Arc offers a range of features to facilitate the automation of hybrid resource management. In this section, we will explore two significant capabilities of Azure Arc related to automation: automatic deployment of changes to multiple resources and the utilization of tags for bulk deployment and changes.

Deploying Changes Automatically to Multiple Resources: One of the primary advantages of Azure Arc is its ability to manage resources across diverse environments through a unified control plane. This includes the automation of change deployment across all resources, regardless of their geographical location. This capability proves particularly valuable when overseeing large-scale hybrid environments that require the consistent deployment of changes across numerous

resources. Azure Arc provides several tools for automating change deployment. Among these, Azure Automation stands out as a key resource, offering a centralized location for managing automation scripts and workflows. Azure Automation enables administrators to define scripts that automatically deploy changes to multiple resources, including those located outside of the Azure ecosystem.

Another essential tool for automating change deployment is Azure Resource Manager (ARM) templates. ARM templates provide a declarative approach to defining infrastructure as code, enabling the automation of change deployment to various resources such as virtual machines, containers, and Kubernetes clusters. Additionally, ARM templates facilitate resource management across multiple environments, encompassing on-premises setups and other public clouds.

**Using Tags for Bulk Deployment/Changes:** Azure Arc's ability to manage resources across diverse environments, including on-premises, cloud, and edge locations, from a single interface proves highly efficient and provides superior resource control. However, automating deployment and changes across all these environments can pose challenges. In this section, we will explore how Azure Arc leverages tags for streamlined bulk deployment and changes. Tags serve as powerful categorization tools in Azure, allowing resources to be grouped based on various criteria such as function, owner, or environment. In Azure Arc, tags facilitate resource management across different environments by enabling the identification of specific resources through their assigned tags.

Utilizing tags for bulk deployment and changes greatly simplifies resource management within Azure Arc. For instance, tags can be employed to deploy a specific set of resources to a particular environment or region, or to update or delete resources in bulk. This proves particularly valuable when dealing with numerous resources scattered across multiple environments. To leverage tags for bulk deployment and changes, Azure Resource Manager (ARM) templates are utilized. ARM templates are declarative files that define the infrastructure and configuration of Azure resources. They enable the automation of resource deployment, configuration, updates, and deletions in a streamlined manner.

One advantage of using ARM templates with tags is the ability to deploy resources to specific environments based on their assigned tags. For example, an ARM template can be created to deploy resources tagged with "Environment: Production" to a production environment, while those tagged with "Environment: Test" are deployed to a testing environment. This simplifies resource management across multiple environments using a single template.

In addition to deployment, tags can also be leveraged for bulk updates and deletions of resources. By utilizing tags to identify a specific set of resources requiring updates or deletion, an ARM template can be employed to carry out the necessary changes. This significantly reduces the time and effort required compared to manual updating or deletion of resources on an individual basis. Another benefit of using tags for automation in Azure Arc is the ability to define policies based on tags. Azure Policy, a service within Azure, enables the enforcement of organizational standards and compliance by establishing rules and restrictions on resources. By utilizing tags to identify resources, policies can be created to apply to specific groups of resources based on their assigned tags. For instance, a policy can be defined to mandate encryption or a specific security configuration for all resources tagged with "Environment: Production.". The use of tags for bulk deployment and changes serves as a powerful automation tool within Azure Arc. It enables efficient resource management across multiple environments, significantly reducing manual effort and time consumption. Furthermore, the ability to establish policies based on tags provides enhanced control and compliance across all environments.

## 5.5 Seamless Integration and Management

Azure Arc offers a wide range of capabilities that facilitate the seamless integration and management of resources in hybrid and multi-cloud environments. Let's explore how Azure Arc can assist us in various aspects:

**Centralized Management:** Azure Arc empowers us with a centralized platform to manage and monitor resources across diverse environments, including on-premises, multi-cloud, and edge locations. The Azure portal acts as a unified control plane, providing a comprehensive view of all resources[72].

**Resource Extension:** By utilizing Azure Arc, we can extend the reach of Azure services, such as Azure Policy, Azure Monitor, and Azure Security Center, to resources located outside of Azure. This extension allows us to enforce consistent management capabilities and policies across our entire infrastructure[73].

**Application Modernization:** Azure Arc facilitates the modernization of existing applications running outside of Azure. By deploying Azure Arc-enabled Kubernetes clusters, we can leverage Azure Kubernetes Service (AKS) features, such as scaling, deployment, and monitoring, for our on-premises or edge workloads. This integration bridges the gap between cloud-native capabilities and existing applications[74].

**Data Services:** Azure Arc extends Azure data services, including Azure SQL Managed Instance and Azure PostgreSQL Hyperscale, to on-premises or multi-cloud environments[75]. This enables us to benefit from Azure's fully managed and scalable data services, simplifying data management and enabling hybrid scenarios.

**Policy Enforcement:** Azure Arc empowers us to enforce Azure policies and governance controls consistently across hybrid and multi-cloud environments. By leveraging Azure Policy, we can define and enforce compliance rules, ensuring adherence to security and regulatory requirements throughout our infrastructure. For instance, we can implement policies to ensure that virtual machines adhere to the Azure compute security baseline and that Windows web servers utilize secure communication protocols[76].

**Hybrid Identity and Security:** Azure Arc seamlessly integrates with Azure Active Directory connector, enabling us to maintain consistent identity and access management practices across hybrid and multi-cloud environments. Additionally, it allows us to leverage Azure Security Center for unified security monitoring, threat detection, and compliance assessment across all resources.

## 5.6 Capacity for hosting resources

Azure Arc enables us to host and manage resources across hybrid and multi-cloud environments. While Azure Arc itself does not provide direct hosting capabilities, it serves as a control plane



and management layer for resources deployed in different environments. Here's how Azure Arc enables resource hosting:

**On-Premises Hosting:** With Azure Arc, we can bring on-premises resources, such as servers, virtual machines, and Kubernetes clusters, under the management umbrella of Azure. By installing the Azure Arc agent on these resources, we can connect them to Azure and manage them through the Azure portal, leveraging Azure services and management capabilities[77].

**Multi-Cloud Hosting:** Azure Arc allows us to extend Azure management and services to resources deployed in other public cloud providers, such as AWS and Google Cloud Platform[78]. By deploying Azure Arc-enabled Kubernetes clusters in these cloud environments, we can manage and monitor them using Azure tools and services as we did in previous chapter with AWS and on-prem machines only.

**Edge Hosting:** Azure Arc supports hosting resources at the edge, bringing Azure services and management capabilities to edge locations. We can deploy Azure Arc-enabled Kubernetes clusters or Azure Arc-enabled data services to edge devices [79] or edge datacenters, enabling consistent management and data processing closer to where it's needed. It was not in scope to cover in this work.

**Extensibility:** Azure Arc enables us to extend Azure services and capabilities to hosted resources. For example, you can extend Azure Policy, Azure Monitor, and Azure Security Center to enforce governance, monitor performance, and enhance security across these resources. It was not in scope to cover in this work.

## 5.7 Cost Efficiency and Utilization

As organizations embrace cloud technologies, managing costs becomes a crucial aspect of cloud operations, especially in hybrid cloud environments. Azure Arc offers organizations a robust platform for cost-efficiently managing and optimizing their cloud resources. Let's refine the text while considering the given instructions:

Cost Efficiency: Azure Arc empowers organizations to centrally manage and monitor their cloud resources across diverse providers and on-premises data centers. This centralized approach reduces the need for separate resource management teams, leading to significant cost savings[80]. Azure Arc also enables organizations to create and enforce resource policies, ensuring efficient and cost-effective resource utilization. A key factor contributing to cost efficiency in Azure Arc is its ability to leverage existing infrastructure and tools. For instance, organizations can use Azure Arc to manage Kubernetes clusters running on any infrastructure, including on-premises or other cloud providers, using familiar tools and processes already in use for Azure Kubernetes Service (AKS) clusters[81]. This eliminates the need for investing in new tools and processes, saving both costs and time.

Effective Resource Utilization for Cost Savings: Azure Arc allows organizations to leverage their existing infrastructure and tools, optimizing resource usage and generating cost savings. For example, organizations can manage and monitor Kubernetes clusters across different infrastructures through Azure Arc's unified view, enabling efficient resource utilization and minimizing the need for overprovisioning. Azure Arc also supports the implementation of policies to optimize resource usage[73]. Organizations can set policies to automatically shut down or scale down unused resources, reducing unnecessary consumption and generating cost savings. Additionally, policies can ensure that resources are provisioned only when necessary, avoiding overprovisioning and maximizing resource efficiency.

In summary, Azure Arc provides organizations with the tools and capabilities to achieve cost efficiency and resource optimization in their cloud operations. By leveraging existing infrastructure and tools, organizations can avoid additional investments and use familiar processes to manage their resources. Implementing resource policies allows for effective utilization and enforcement of best practices, resulting in cost savings. Utilizing tags enables efficient resource management and allocation based on specific criteria. With Azure Arc, organizations can maximize the value of their cloud resources while minimizing costs, ultimately optimizing their overall cloud operations.

# CHAPTER 6

## 6 Conclusion

Azure Arc, a comprehensive solution for managing and securing resources in hybrid and multi-cloud environments, offers a multitude of best practices and addresses various challenges in hybrid cloud management. This platform combines perplexity and burstiness in its writing style, mimicking the human-like approach of blending sentence complexity and length.

In response to Research Question 1, the best practices of using Azure Arc for hybrid cloud management revolve around policy enforcement, automation, resource optimization, and integration. Enforcing policies on Azure Arc-enabled resources ensures compliance with organizational policies and regulations. By creating policies in Azure Policy and assigning them to resources, IT teams can simplify policy management and improve security. Automation is another crucial best practice, accomplished through tools like Azure Automation and ARM templates. Automating changes across multiple environments streamlines management and reduces errors, enhancing operational efficiency. Resource optimization, achieved by leveraging existing infrastructure investments and implementing policies for efficient resource utilization, helps organizations achieve cost efficiency. Furthermore, Azure Arc simplifies resource management and integration by providing a unified control plane and application modernization capabilities. This centralized approach enables consistent management across on-premises, multi-cloud, and edge locations, eliminating the need for multiple management tools and interfaces.

Addressing Research Question 2, the use of Azure Arc and its related services significantly impacts the security of a hybrid cloud environment while providing a high level of control for the end-user. By enforcing policies on Azure Arc-enabled resources, organizations can enhance their security posture and reduce the potential attack surface for security threats. Compliance with organizational policies and regulations is ensured, contributing to a secure hybrid cloud environment. Azure Arc integrates with Azure Active Directory, enabling centralized access control and creating a consistent security framework. The end-user achieves granular control over resource management, access control, and policy enforcement, empowering them to tailor security measures according to their specific requirements. Azure Arc's unified control plane facilitates a comprehensive view of resources and their security status, enabling effective monitoring and incident response. However, it is important to acknowledge the challenges involved. Managing resources across diverse environments, ensuring consistent policy enforcement, and adapting to the learning curve associated with Azure Arc implementation can pose difficulties. Integrating existing on-premises infrastructure with Azure services requires careful planning. Consistent policy enforcement across hybrid environments requires thorough testing and monitoring. Additionally, optimizing resource utilization and cost efficiency requires continuous evaluation and adjustment.

In conclusion, Azure Arc provides a comprehensive and adaptable solution for managing hybrid cloud environments, addressing key challenges, and following best practices. With its robust features for policy enforcement, automation, resource optimization, and integration, Azure Arc enables organizations to achieve compliance, enhance security, and improve operational efficiency. By leveraging Azure Arc, end-users gain extensive control over resource management and access control, strengthening hybrid cloud security. However, it is important to acknowledge and navigate challenges such as the complexity of resource management, ensuring consistency in policy enforcement, and the learning curve associated with implementing Azure Arc. By carefully considering these factors and utilizing the capabilities of Azure Arc, organizations can successfully manage their hybrid cloud environments and fully leverage their advantages.



# List of Figures

|   |    |
|---|----|
| Figure 1.1: Hybrid Cloud integration with Azure Arc.....                  | 2  |
| Figure 4.1: Installing azure arc agent on AWS instance.....               | 30 |
| Figure 4.2 : Azure Portal credentials. ....                               | 31 |
| Figure 4.3: Listed AWS instances on Azure portal .....                    | 31 |
| Figure 4.4: Adding node with microk8s .....                               | 33 |
| Figure 4.5: microk8s installation .....                                   | 33 |
| Figure 4.6: Verifying the created nodes.....                              | 34 |
| Figure 4.7: logging in with azure-cli .....                               | 34 |
| Figure 4.8: creating a service principal.....                             | 35 |
| Figure 4.9: Pre-requisite registration for Azure Arc.....                 | 35 |
| Figure 4.10: Sample deployment in K8s cluster.....                        | 36 |
| Figure 4.11: Environment variables for later user .....                   | 37 |
| Figure 4.12: Using connected k8s command to connect Azure Arc.....        | 37 |
| Figure 4.13: service account and role binding creation .....              | 38 |
| Figure 4.14: Service account token to access connected cluster. ....      | 38 |
| Figure 4.15: secret.yaml creation.....                                    | 39 |
| Figure 4.16: fetching secret .....  | 39 |
| Figure 4.17: Overview of connected on-prem cluster resources. ....        | 40 |
| Figure 4.18: AKS creation Page .....                                      | 42 |
| Figure 4.19: Running Script to register AKS with Azure Arc.....           | 43 |
| Figure 4.20: Service account, Clusterrolebinding and Secret creation..... | 44 |
| Figure 4.25: Overview of connected AKS cluster resources.....             | 44 |
| Figure 4.22: User creation for EKS .....                                  | 46 |
| Figure 4.23: Creating aws-cli access for ubuntu server.....               | 47 |
| Figure 4.24: Access key.....  | 48 |
| Figure 4.25: Logging in to aws environment with aws-cli.....              | 48 |
| Figure 4.26: Assigning IAM roles to eks-user.....                         | 49 |
| Figure 4.27: EKS creation page.....                                       | 50 |
| Figure 4.28: Node group.....  | 51 |
| Figure 4.29: Workload on EKS-Node .....                                   | 51 |

|   |    |
|---|----|
| Figure 4.30: Overview of EKS Cluster on Ubuntu.....               | 52 |
| Figure 4.31: Using Connectedk8s to onboard EKS cluster. ....      | 52 |
| Figure 4.32: Overview of connected EKS cluster resources .....    | 53 |
| Figure 4.33: On-Prem, AKS and EKS integrated with Azure Arc ..... | 53 |
| Figure 4.34: Update the terraform version.....                    | 54 |
| Figure 4.35: Register the resource providers. ....                | 55 |
| Figure 4.36: Access key creation.....                             | 55 |
| Figure 4.37: SQL Server Management Studio Installation.....       | 57 |
| Figure 4.38: SQL Server Management Studio.....                    | 57 |
| Figure 4.39: Script to connect Arc-enabled SQL to Azure Arc.....  | 58 |
| Figure 4.40: SQL Machine and extensions.....                      | 58 |
| Figure 5.1: Azure Arc capabilities.....                           | 60 |

# List of Tables

|   |    |
|---|----|
| Table 2.1: Different cloud computing models and their comparison [15].....                                  | 7  |
| Table 2.2: Challenges and solutions associated with effective workload management in hybrid platforms. .... | 19 |



# Appendix

## A. Implementation codes for VMs and SQL servers

### A.1 Onboarding script

The script used for onboarding of the AWS instance:[\[56\]](#)

```
try {
    $env:SUBSCRIPTION_ID = "";

    $env:RESOURCE_GROUP = "HCW-RG";

    $env:TENANT_ID = "";

    $env:LOCATION = "westeurope";

    $env:AUTH_TYPE = "token";

    $env:CORRELATION_ID = "dc425058-3946-435d-928b-01e7ecb96d92";

    $env:CLOUD = "AzureCloud";

    [Net.ServicePointManager]::SecurityProtocol =
[Net.ServicePointManager]::SecurityProtocol -bor 3072;

    Invoke-WebRequest -UseBasicParsing -Uri "https://aka.ms/azcmagent-windows" -
TimeoutSec 30 -OutFile "$env:TEMP\install_windows_azcmagent.ps1";

    & "$env:TEMP\install_windows_azcmagent.ps1";

    if ($LASTEXITCODE -ne 0) { exit 1;}

    & "$env:ProgramW6432\AzureConnectedMachineAgent\azcmagent.exe" connect --
resource-group "$env:RESOURCE_GROUP" --tenant-id "$env:TENANT_ID" --location
"$env:LOCATION" --subscription-id "$env:SUBSCRIPTION_ID" --cloud "$env:CLOUD" --
correlation-id "$env:CORRELATION_ID";
}
```

```

catch {
    $logBody =
    @{"subscriptionId"=$env:SUBSCRIPTION_ID";resourceGroup="$env:RESOURCE_GROUP
";tenantId="$env:TENANT_ID";location="$env:LOCATION";correlationId="$env:CORRE
LATION_ID";authType="$env:AUTH_TYPE";operation="onboarding";messageType=$_.
FullyQualifiedErrorId;message="$_.Exception"};

    Invoke-WebRequest -UseBasicParsing -Uri "https://gbl.his.arc.azure.com/log" -Method
    "PUT" -Body ($logBody | ConvertTo-Json) | out-null;

    Write-Host -ForegroundColor red $_.Exception;
}

```

## A.2 Terraform Code

The following terraform code is to deploy aws instance. [\[69\]](#)

```

provider "aws" {
    region = "eu-west-1"
}
#Create an EC2 instance
resource "aws_instance" "example" {
    ami          = "ami-0c94855ba95c71c99"
    instance_type = "t2.micro"
    tags = {
        Name = "example-instance"
    }
    # Create a security group that allows SSH and SQL Server connections
    vpc_security_group_ids = [aws_security_group.example.id]
    # Attach an IAM role that allows access to S3 and RDS
    iam_instance_profile = aws_iam_instance_profile.example.name
}
# Create a security group that allows SSH and SQL Server connections
resource "aws_security_group" "example" {
    name_prefix = "example-"
}

```

```

ingress {
  from_port = 22
  to_port   = 22
  protocol  = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
}
ingress {
  from_port = 1433
  to_port   = 1433
  protocol  = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
}
}

```

### A.3 SQL installation Script

The following script is for installing SQL instance. [\[71\]](#)

```

param ($servicePrincipalAppId, $servicePrincipalTenantId, $servicePrincipalSecret)

# These settings will be replaced by the portal when the script is generated
$subId = ""
$tenantId = ""
$resourceGroup = "HCW-RG"
$location = "westeurope"
$proxy=""
$licenseType="LicenseOnly"

# These optional variables can be replaced with valid service principal details
# if you would like to use this script for a registration at scale scenario, i.e. run it on
multiple machines remotely
# For more information, see https://docs.microsoft.com/sql/sql-server/azure-arc/connect-at-scale
#
# For security purposes, passwords should be stored in encrypted files as secure strings
#
#$servicePrincipalAppId = ""
#$servicePrincipalTenantId = ""
#$servicePrincipalSecret = ""

```

```

$currentDir = Get-Location
$unattended = $servicePrincipalAppld -And $servicePrincipalTenantId -And
$servicePrincipalSecret

[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12

try {
    if ($proxy) {
        Invoke-WebRequest -Proxy $proxy -Uri https://aka.ms/AzureExtensionForSQLServer -
        OutFile AzureExtensionForSQLServer.msi
    } else {
        Invoke-WebRequest -Uri https://aka.ms/AzureExtensionForSQLServer -OutFile
        AzureExtensionForSQLServer.msi
    }
} catch {
    throw "Invoke-WebRequest failed: $_"
}

try {
    $exitcode = (Start-Process -FilePath msixexec.exe -ArgumentList @("/i",
    "AzureExtensionForSQLServer.msi", "/!v", "installationlog.txt", "/qn") -Wait -
    Passthru).ExitCode

    if ($exitcode -ne 0) {
        $message = "Installation failed: Please see $currentDir\installationlog.txt file for more
        information."
        Write-Host -ForegroundColor red $message
        return
    }

    if ($unattended) {
        & "$env:ProgramW6432\AzureExtensionForSQLServer\AzureExtensionForSQLServer.exe" -
        -subld $subld --resourceGroup $resourceGroup --location $location --tenantid
        $servicePrincipalTenantId --service-principal-app-id $servicePrincipalAppld --service-
        principal-secret $servicePrincipalSecret --proxy $proxy --licenseType $licenseType
    } else {
        & "$env:ProgramW6432\AzureExtensionForSQLServer\AzureExtensionForSQLServer.exe" -
        -subld $subld --resourceGroup $resourceGroup --location $location --tenantid $tenantId --
        proxy $proxy --licenseType $licenseType
    }

    if($LASTEXITCODE -eq 0){
        Write-Host -ForegroundColor green "Azure extension for SQL Server is successfully installed.
        If one or more SQL Server instances are up and running on the server, Arc-enabled SQL

```

```

Server instance resource(s) will be visible within a minute on the portal. Newly installed
instances or instances started now will show within an hour."
}
else{
$message = "Failed to install Azure extension for SQL Server. Please see
$currentDir\AzureExtensionForSQLServerInstallation.log file for more information."
Write-Host -ForegroundColor red $message
}
}
catch {
Write-Host -ForegroundColor red $_.Exception
throw
}

```

## A.4 Deployment.yaml

The following yaml file is to deploy a sample workload in K8s Cluster [\[63\]](#)

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 80

```

## A.5 Bash Script

Bash script to speed up deployment for AKS [\[65\]](#)

```

echo " <--> Exporting environment variables for later use in script <---> "
export serviceapp="<appID>"
export pwd="<service App password>"

```

```

export tenant="<<tenantID>"
export rg_arc="rg-Arc"
export Azure_arc_ClusterName="AKS-Azure"
export AKS_name="aks-demo"
export AKS_rg="asim-devbox-05jul-05sept"
echo "exported env variables"
echo " <-----> Using Service principle to Log in to Azure <-----> "
az login --service-principal --username $serviceapp --password $pwd --tenant $tenant
echo " <-----> setting up for Azure Arc providers <-----> "
az provider register --namespace Microsoft.Kubernetes --wait
az provider register --namespace Microsoft.KubernetesConfiguration --wait
az provider register --namespace Microsoft.ExtendedLocation --wait
az provider show -n Microsoft.Kubernetes -o table
az provider show -n Microsoft.KubernetesConfiguration -o table
az provider show -n Microsoft.ExtendedLocation -o table
# Getting AKS credentials
echo " <-----> Getting AKS credentials into kubeconfig file <-----> "
az aks get-credentials --name $AKS_name \
                        --resource-group $AKS_rg --overwrite-existing
kubectl config use-context aks-demo
echo " <-> Using connectedk8s extention to Connect the cluster to Azure Arc <-> "
az connectedk8s connect --name $Azure_arc_ClusterName \
--resource-group $rg_arc --tags 'AKS'

```

# Bibliography

1. Nagarajan, G. and K. Sampath Kumar, *Security Threats and Challenges in Public Cloud Storage*, in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*. 2021. p. 97-100.
2. Linthicum, D.S., *Emerging hybrid cloud patterns*. IEEE Cloud Computing, 2016. 3(1): p. 88-91.
3. Flexera. *Flexera Releases 2020 State of the Cloud Report*. 2020; Available from: [www.flexera.com/about-us/press-center/flexera-releases-2020-state-of-the-cloud-report.html](http://www.flexera.com/about-us/press-center/flexera-releases-2020-state-of-the-cloud-report.html).
4. Benhssayen, K. and A. Ettalbi, *Semantic interoperability framework for IAAS resources in multi-cloud environment*. International Journal of Computer Science & Network Security, 2021. 21(2): p. 1-8.
5. Hr, S. and T. S., *A Hybrid Cloud Approach for Efficient Data Storage and Security*, in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*. 2021. p. 1072-1076.
6. Microsoft. *Azure Arc overview*. 2022; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/overview>.
7. Microsoft. *Manage configurations for Azure Arc-enabled servers*. Available from: <https://learn.microsoft.com/en-us/azure/architecture/hybrid/azure-arc-hybrid-config>.
8. Kumar, R., *Bring Azure data services to your infrastructure with Azure Arc*. 2019, Microsoft.
9. Sanders, C., *Microsoft Cloud for Sovereignty: The most flexible and comprehensive solution for digital sovereignty*. 2022, Microsoft.
10. Qarawlus, H., et al., *Sovereign Data Exchange in Cloud-Connected IoT using International Data Spaces*, in *2021 IEEE Cloud Summit (Cloud Summit)*. 2021. p. 13-18.
11. Microsoft. *What are managed identities for Azure resources?* 2023; Available from: <https://learn.microsoft.com/en-us/azure/active-directory/managed-identities-azure-resources/overview>.
12. Microsoft. *Authenticate against Azure resources with Azure Arc-enabled servers*. 2021; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/servers/managed-identity-authentication>.
13. *Amazon Web Services OutPost*. Available from: <https://aws.amazon.com/outposts/>.
14. *IBM Cloud*. Available from: <https://www.ibm.com/cloud>.
15. Guruh Aryotejo, D.Y.K., Mufadhol, *Hybrid cloud: bridging of private and public cloud computing*. Journal of Physics: Conference Series, 2018.
16. Rashid, A. and A. Chaturvedi, *Cloud Computing Characteristics and Services A Brief Review*. International Journal of Computer Sciences and Engineering, 2019. 7(2): p. 421-426.

17. Van den Bossche, R., K. Vanmechelen, and J. Broeckhove, *Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds*. Future Generation Computer Systems, 2013. 29(4): p. 973-985.
18. Roy, N., A. Dubey, and A. Gokhale, *Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting*, in *2011 IEEE 4th International Conference on Cloud Computing*. 2011. p. 500-507.
19. Bal, P.K., et al., *A Joint Resource Allocation, Security with Efficient Task Scheduling in Cloud Computing Using Hybrid Machine Learning Techniques*. Sensors (Basel), 2022. 22(3).
20. Rashidi, B., M. Sharifi, and T. Jafari, *A Survey on Interoperability in the Cloud Computing Environments*. International Journal of Modern Education and Computer Science, 2013. 5(6): p. 17-23.
21. Gonçalves, V. and P. Ballon, *Adding value to the network: Mobile operators' experiments with Software-as-a-Service and Platform-as-a-Service models*. Telematics Informatics, 2011. 28: p. 12-21.
22. Andrew Joint, E.B., *Knowing the past to understand the present1 – issues in the contracting for cloud based services*. Computer Law & Security Review, 2011. 27(4): p. 407-415.
23. Birje, M.N., et al., *Cloud computing review: concepts, technology, challenges and security*. International Journal of Cloud Computing, 2017. 6(1).
24. Amazon Web Services (AWS). (n.d.). Available from: <https://aws.amazon.com/>.
25. Microsoft. *Microsoft Azure*. Available from: <https://azure.microsoft.com/>.
26. Google Cloud Platform (GCP). Available from: <https://cloud.google.com/>.
27. Oracle. *Oracle Cloud*. Available from: <https://www.oracle.com/cloud/>.
28. Hong, J., et al., *An Overview of Multi-cloud Computing*, in *Web, Artificial Intelligence and Network Applications*. 2019. p. 1055-1068.
29. Ghanam, Y., J. Ferreira, and F. Maurer, *Emerging Issues & Challenges in Cloud Computing—A Hybrid Approach*. Journal of Software Engineering and Applications, 2012. 05(11): p. 923-937.
30. Zhang, Q., L. Cheng, and R. Boutaba, *Cloud computing: state-of-the-art and research challenges*. Journal of Internet Services and Applications, 2010. 1(1): p. 7-18.
31. Sun, Y., et al., *Data Security and Privacy in Cloud Computing*. International Journal of Distributed Sensor Networks, 2014. 10(7).
32. VMware vSphere. Available from: <https://www.vmware.com/products/vsphere.html>.
33. OpenStack. Available from: <https://www.openstack.org/>.
34. Microsoft Azure Stack. Available from: <https://azure.microsoft.com/en-us/overview/azure-stack/>.
35. IBM Cloud Private. Available from: <https://www.ibm.com/cloud/private>.
36. Nutanix Enterprise Cloud. Available from: <https://www.nutanix.com/what-we-do>
37. Pramod, N., Muppalla, A. K., & Srinivasa, *Limitations and Challenges in Cloud-Based Applications Development*. . 10.1007/978-1-4471-5031-2\_3. , 2013.
38. Microsoft. (n.d.). *Azure Stack Hub*. Available from: <https://azure.microsoft.com/en-us/products/azure-stack/hub/>.
39. Google Cloud. (n.d.). *Anthos*; Available from: <https://cloud.google.com/anthos/>.



40. IBM Cloud. (n.d.). IBM Cloud Satellite. Available from: <https://www.ibm.com/cloud/satellite>.
41. VMware. (n.d.). VMware Cloud on AWS. Available from: <https://cloud.vmware.com/vmc-aws>.
42. Khan, S.U. and N. Ullah, *Challenges in the adoption of hybrid cloud: an exploratory study using systematic literature review*. The Journal of Engineering, 2016. 2016(5): p. 107-118.
43. Kang, C., et al., *Complex Service Management in a Hybrid Cloud*, in *2011 Annual SRII Global Conference*. 2011. p. 34-46.
44. Mazhelis, O. and P. Tyrvaïnen, *Role of Data Communications in Hybrid Cloud Costs*, in *2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications*. 2011. p. 138-145.
45. Chen, D. and H. Zhao, *Data Security and Privacy Protection Issues in Cloud Computing*, in *2012 International Conference on Computer Science and Electronics Engineering*. 2012. p. 647-651.
46. Nzanywayingoma, F. and Y. Yang, *Efficient resource management techniques in cloud computing environment: a review and discussion*. International Journal of Computers and Applications, 2018. 41(3): p. 165-182.
47. Ballani, H., et al., *Towards predictable datacenter networks*. SIGCOMM Comput. Commun. Rev., 2011. 41(4): p. 242-253.
48. D. Gmach, J.R., L. Cherkasova, G. Belrose, T. Turicchi and A. Kemper,, *An integrated approach to resource pool management: Policies, efficiency and quality metrics*. IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN), Anchorage, AK, USA, 2008, 2008: p. 326-365.
49. Hu, L.R., Kyung & Silva, M. & Schwan, Karsten., *Flexible Group Resource Offerings in Clouds. Proceedings*. International Conference on Distributed Computing Systems - 10.1109/ICDCS.2012.61. , 2012: p. 406-415.
50. Armbrust, M., et al., *A view of cloud computing*. Communications of the ACM, 2010. 53(4): p. 50-58.
51. *Best Cloud Management Platforms*. Available from: <https://www.g2.com/categories/cloud-management-platforms>.
52. Gotor, F.R. *Plug into simplicity with Tietoevry Sovereign Cloud*. Available from: <https://www.tietoevry.com/en/tech-services/cloud-and-infrastructure/sovereign-cloud>.
53. Microsoft. *Plan and deploy Azure Arc-enabled servers*. 2022; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/servers/plan-at-scale-deployment>.
54. Microsoft. *Monitor a hybrid machine with VM insights*. 2022; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/servers/learn/tutorial-enable-vm-insights>.
55. Pathak, R.C. and P. Khandelwal, *A Model for Hybrid Cloud Integration: With a Case Study for IT Service Management (ITSM)*, in *2017 IEEE International Conference on Cloud Computing in Emerging Markets (CEM)*. 2017. p. 113-118.
56. Microsoft. *Connect your non-Azure machines to Microsoft Defender for Cloud with Azure Arc*. 2023; Available from: <https://learn.microsoft.com/en-us/azure/defender-for-cloud/quickstart-onboard-machines?pivots=azure-arc>.
57. Ubuntu, C.-. Available from: <https://ubuntu.com/>.

58. Microsoft. *Application and service principal objects in Azure Active Directory*. 2022; Available from: <https://learn.microsoft.com/en-us/azure/active-directory/develop/app-objects-and-service-principals>.
59. Multipass, C.-. Available from: <https://multipass.run/>.
60. BRÜNDL, T. *MicroK8s Multinode*. 2022; Available from: <https://it-infrastructure.solutions/azure-arc-enabled-microk8s-multinode-cluster/>.
61. MicroK8s, C.-. *The lightweight Kubernetes*. Available from: <https://microk8s.io/>.
62. Azure Arc Jumpstart. Available from: [https://azurearcjumpstart.io/azure\\_arc\\_jumpstart/azure\\_arc\\_k8s/general/onboard\\_k8s/](https://azurearcjumpstart.io/azure_arc_jumpstart/azure_arc_k8s/general/onboard_k8s/).
63. Kubernetes. *Run a Stateless Application Using a Deployment*. Available from: <https://kubernetes.io/docs/tasks/run-application/run-stateless-application-deployment/>.
64. Microsoft. *Quickstart: Deploy an Azure Kubernetes Service (AKS) cluster using the Azure portal*. 2023; Available from: <https://learn.microsoft.com/en-us/azure/aks/learn/quick-kubernetes-deploy-portal?tabs=azure-cli>.
65. Microsoft. *Azure Kubernetes Service (AKS) hybrid*. Available from: [https://azurearcjumpstart.io/azure\\_arc\\_jumpstart/azure\\_arc\\_k8s/aks/](https://azurearcjumpstart.io/azure_arc_jumpstart/azure_arc_k8s/aks/).
66. Amazon. *Amazon Elastic Kubernetes Service (EKS)*. Available from: <https://aws.amazon.com/eks/>.
67. Amazon. *Installing or updating the latest version of the AWS CLI*. Available from: <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>.
68. Microsoft. *Configure Terraform in Azure Cloud Shell with Bash*. 2023; Available from: <https://learn.microsoft.com/en-us/azure/developer/terraform/get-started-cloud-shell-bash?tabs=bash>.
69. MakendranG. *Creating an EC2 using the Terraform configuration files*. 2022; Available from: <https://dev.to/aws-builders/how-to-create-an-ec2-instance-on-aws-using-terraform-4lg>.
70. MICROSOFT. Available from: [https://github.com/microsoft/azure\\_arc/blob/main/azure\\_arc\\_sqlsrv\\_jumpstart/aws/winsrv/terraform/scripts/sql.ps1.tmpl](https://github.com/microsoft/azure_arc/blob/main/azure_arc_sqlsrv_jumpstart/aws/winsrv/terraform/scripts/sql.ps1.tmpl).
71. Microsoft. *Create an Azure service principal with Azure PowerShell*. 2023; Available from: <https://learn.microsoft.com/en-us/powershell/azure/create-azure-service-principal-azureps?view=azps-10.0.0>.
72. Microsoft. *Azure Monitor overview*. 2023; Available from: <https://learn.microsoft.com/en-us/azure/azure-monitor/overview>.
73. Microsoft. *Azure Policy built-in definitions for Azure Arc-enabled servers*. 2023; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/servers/policy-reference>.
74. Microsoft. *Azure Arc enabled Kubernetes preview and new ecosystem partners*. 2023; Available from: <https://azure.microsoft.com/en-us/blog/azure-arc-enabled-kubernetes-preview-and-new-ecosystem-partners/>.
75. Microsoft. *What are Azure Arc-enabled data services?* 2023; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/data/overview>.

76. Microsoft. *Apply Azure Policy to Azure Arc-enabled Servers in a Hybrid environment*. 2022; Available from: <https://techcommunity.microsoft.com/t5/itops-talk-blog/apply-azure-policy-to-azure-arc-enabled-servers-in-a-hybrid/ba-p/3540827>.
77. Microsoft. *Overview of Azure Connected Machine agent*. Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/servers/agent-overview>.
78. Nguyen, N.X. *Hybrid and Multi-Cloud Management with Azure Arc*. 2022; Available from: <https://adamtheautomator.com/azure-arc/>.
79. Soni, J. *Azure Arc: Manage Your Resources Across Multiple Clouds, Data Centers, and Edge Locations*. 2023; Available from: <https://www.linkedin.com/pulse/azure-arc-manage-your-resources-across-multiple-clouds-jaimin-soni/>.
80. Microsoft. *Cost optimization in a hybrid workload*. 2022; Available from: <https://learn.microsoft.com/en-us/azure/well-architected/hybrid/hybrid-cost>.
81. Microsoft. *What is Azure Arc-enabled Kubernetes?* 2023; Available from: <https://learn.microsoft.com/en-us/azure/azure-arc/kubernetes/overview>.