



**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

To install azure CLI tools to list all the resources and managing vm’s.

Name: Subiksha R Department:AML



**INTRODUCTION:**

**OVERVIEW: Top of Form**

To effectively manage virtual machines (VMs) and list their resources for a proof of concept (PoC), the installation and configuration of appropriate command-line interface (CLI) tools are crucial. CLI tools provide a direct, scriptable way to interact with virtualized environments, offering flexibility, automation, and streamlined workflows. These tools allow users to manage tasks such as provisioning, modifying, monitoring, and troubleshooting VMs, all from the terminal or a script, reducing the need for graphical user interfaces (GUIs) and enabling more efficient resource management.

When working with cloud-based VM environments, popular platforms like Amazon Web Services (AWS), Microsoft Azure, and Google Cloud provide their own dedicated CLI tools such as the AWS CLI, Azure CLI, and Google Cloud SDK. These tools enable users to create and manage VMs, configure networking, handle storage, and query resource usage. For example, AWS CLI allows you to provision EC2 instances, attach EBS volumes, and list running instances, while Azure CLI can be used to interact with Azure Virtual Machines and manage related resources like virtual networks and disks. Similarly, Google Cloud SDK provides commands to interact with Google Compute Engine, making it easier to perform tasks such as scaling, deploying, and monitoring VMs in Google Cloud.

For those managing local or on-premises VMs, CLI tools like vSphere CLI for VMware, VBoxManage for VirtualBox, and virsh for KVM-based environments provide robust control over virtual machines. These tools allow users to automate VM creation, control VM states (e.g., start, stop, suspend), and even retrieve performance metrics such as CPU and memory usage, disk utilization, and network activity. For example, VMware’s vSphere CLI allows administrators to manage ESXi hosts and virtual machines, while VBoxManage lets users automate operations in VirtualBox, including snapshot management and VM configuration changes.

Moreover, when evaluating resources for a PoC, it’s essential to have insight into how the VMs are utilizing CPU, RAM, disk, and network resources. CLI tools allow users to gather this data programmatically, which is particularly useful for performance benchmarking and resource optimization. Commands for resource listing and monitoring can help assess whether the VM setup meets the needs of the PoC or if adjustments are necessary (e.g., scaling VMs up or down based on performance).

Additionally, combining these tools with automation platforms like Terraform or Ansible can further enhance VM management by allowing infrastructure-as-code (IaC) deployments. This is especially useful for PoCs that require rapid provisioning, teardown, and replication of environments across different stages of the project.

Bottom of Form

**OBJECTIVE:**

1. **Simplify VM Management:** To provide a streamlined approach to managing virtual machines (VMs) through CLI tools, reducing reliance on graphical user interfaces (GUIs).

2. **Automate Tasks:** To enable automation of VM provisioning, modification, and monitoring using CLI commands, allowing for more efficient workflows and reduced manual effort.

3. **Support Multiple Environments:** To provide tools that work across various platforms, including cloud providers (AWS, Azure, Google Cloud) and on-premises virtualization technologies (VMware, VirtualBox, KVM).

4. **Monitor Resources:** To facilitate the collection of resource usage data (CPU, RAM, disk, network) through CLI tools for accurate performance monitoring and optimization during a proof of concept (PoC).

5. **Enhance Resource Scaling:** To enable scaling of VMs based on performance insights, ensuring that resources align with the requirements of a PoC.

6. **Promote Infrastructure as Code (IaC):** To integrate CLI tools with automation frameworks like Terraform or Ansible for repeatable and scalable VM management.

7.**Improve Performance Benchmarking:** To use CLI tools for detailed benchmarking and resource usage tracking, helping to optimize VM configurations during PoC development.

**Importance of Setting Up a Local Repository**Bottom of Form

1. Increased Efficiency: Automates repetitive tasks like creating and managing VMs, saving time and effort.

2. Cost Optimization: Helps track and manage resources efficiently, avoiding unnecessary cloud costs and ensuring you only use what’s needed for your PoC.

3. Scalability: Easily scale resources up or down as your PoC requirements change.

4. Consistency: Ensures that VM setups are identical every time, reducing errors and making the environment reproducible.

5. Better Resource Monitoring: Provides insights into how VMs are using CPU, memory, and storage, helping to optimize performance during your PoC.

6. Faster Troubleshooting: Enables quick detection and resolution of issues, minimizing downtime during the PoC.

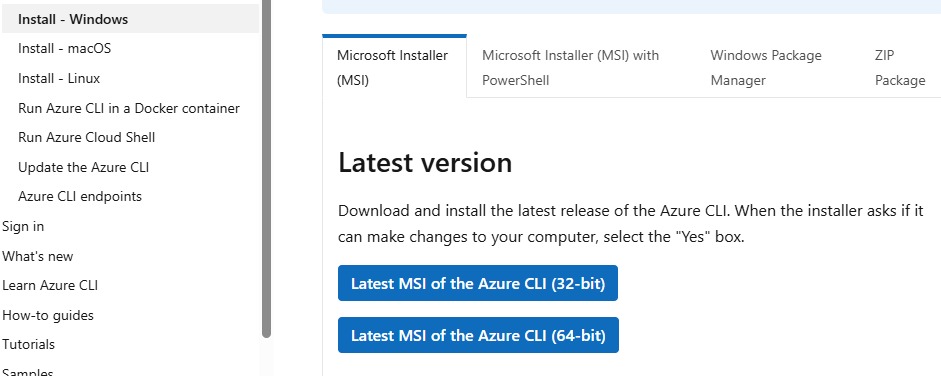
7. Collaboration: Standardizes VM setups across teams, making it easier to share and collaborate on the PoC.

8. Cross-Platform Flexibility: Works across both cloud and on-premises virtual environments, giving you more control and options for managing your infrastructure.

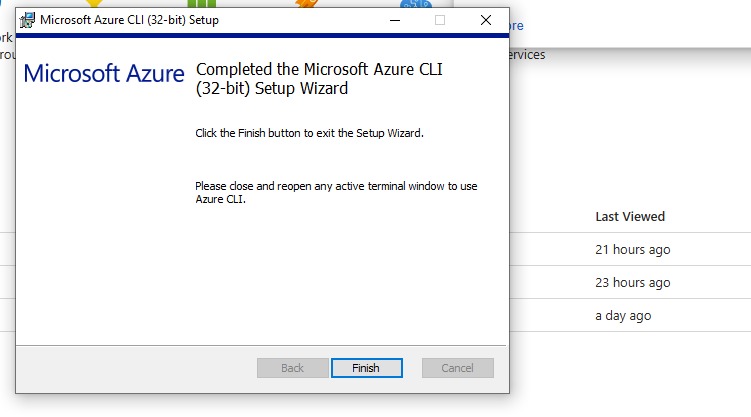
**STEP BY STEP OVERVIEW:**

**STEP 1:**

Install the azure CLI from student guide website.

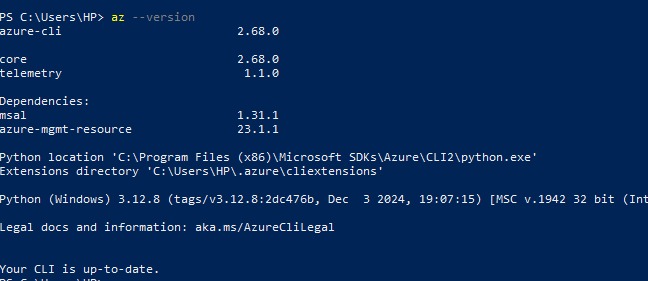


**Click finish to install:**



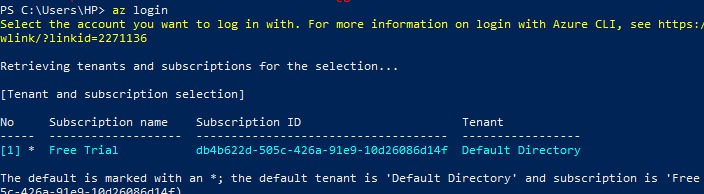
**Step 2:**

**Give az version command to verify the installation:**



**Step 3:**

**Give az login command to login into azure current subscription to access the resources:**

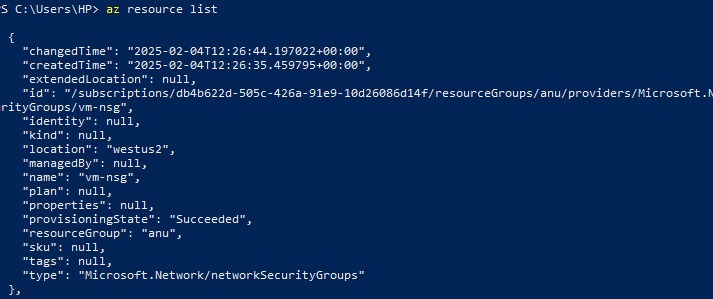




**Step 4:**

**To list the resources created inside this subscription:**





**Step 5:**

**TO start or stop a vm process you can use this command:**





**Expected outcome:**

1. **Faster Setup and Execution**: Quick provisioning, management, and teardown of VMs for efficient testing and iteration in the PoC.

2. **Optimized Resource Usage**: Better control and understanding of resource consumption (CPU, memory, disk), leading to efficient allocation and cost savings.

3. **Scalable Infrastructure**: Ability to scale VMs up or down based on PoC needs, ensuring the right amount of resources are available when required.

4. **Consistent Environments**: Reproducible and standardized VM configurations, ensuring a stable and reliable PoC environment across tests.

5. **Improved Performance Monitoring**: Access to real-time resource metrics that allow for data-driven decisions about resource allocation and optimization.

6. **Reduced Downtime**: Efficient troubleshooting and quick resolution of issues, leading to minimal disruption during the PoC.

7. **Smooth Collaboration**: Seamless collaboration across teams with standardized scripts and environments, ensuring everyone is working with the same setup.

8. **Documentation and Automation**: Automated workflows and scripts that can be reused or shared, enabling faster future deployments and consistent environments for subsequent PoCs.