**Lab 15: Azure Enhanced Security**

**Introduction:**

Security is critical in any cloud environment. Azure offers several services to enhance access security, such as Conditional Access and Privileged Identity Management (PIM) through Microsoft Entra ID, as well as Azure Key Vault for secure storage. This lab will guide you through setting up and using these services to enhance your cloud security posture.

**Objectives:**

* Create and configure Conditional Access policies.
* Set up and use Privileged Identity Management (PIM).
* Use Azure Key Vault to securely store and retrieve secrets.

**Steps:**

**Part 1: Conditional Access Setup**

**Step 1:** Access Microsoft Entra ID

* **Explanation**: Conditional Access policies are configured within Microsoft Entra ID.
* **How to Do It**: In the Azure portal, search for "Microsoft Entra ID" or "Azure Active Directory" in the search bar, then click on it.

**Step 2:** Set Up a Conditional Access Policy

* **Explanation**: Conditional Access policies help enforce security requirements based on user conditions.
* **How to Do It**:
  + Click on "Security" in the left-hand menu.
  + Click on "Conditional Access."
  + Click "New policy" and give the policy a name, like "MFA for Admins."
  + Under "Assignments," define the users or groups to apply the policy to, such as a specific group of administrators.
  + Define the conditions, like requiring Multi-Factor Authentication (MFA) or blocking risky sign-ins.
  + Set the controls for the policy to either allow or block based on the conditions.
  + Click "Create" to save the policy.

**Part 2: Setting Up Privileged Identity Management (PIM)**

**Step 1:** Enable Azure PIM

* **Explanation**: PIM allows you to manage access to critical roles on a just-in-time basis.
* **How to Do It**: In the Microsoft Entra ID page, click on "Privileged Identity Management" in the left-hand menu.

**Step 2:** Onboard a Role for PIM

* **Explanation**: By onboarding a role, you enable PIM to manage and monitor access to that role.
* **How to Do It**:
  + Click on "Azure AD roles."
  + Click "Onboard" to start the onboarding process if you haven't already.
  + Select the specific role you want to manage, like "Global Administrator" or "User Administrator."
  + Click "Next" and follow the prompts to complete the onboarding process.

**Step 3:** Configure PIM Policies

* **Explanation**: You can set up time-bound and approval-required access.
* **How to Do It**:
  + Click on "Settings" for the role you onboarded.
  + Define the maximum duration of access, set up approval workflows, and require justification for access.
  + Click "Save" to apply the settings.

**Part 3: Using Azure Key Vault for Secure Storage**

**Step 1:** Create an Azure Key Vault

* **Explanation**: Key Vault is used to securely store secrets, keys, and certificates.
* **How to Do It**:
  + In the Azure portal, search for "Key Vaults."
  + Click "Create" to create a new Key Vault.
  + Provide a name, subscription, and resource group for the Key Vault.
  + Click "Review + create" and then "Create."

**Step 2:** Add a Secret to the Key Vault

* **Explanation**: Secrets can include passwords, API keys, and other sensitive data.
* **How to Do It**:
  + In the Key Vault, click on "Secrets" in the left-hand menu.
  + Click "Generate/Import" to add a new secret.
  + Provide a name and the value of the secret.
  + Click "Create."

**Step 3:** Enable Managed Identity for Your Compute Resource

* **Explanation**: Managed identity simplifies authentication with Key Vault.
* **How to Do It**:
  + Go to the Azure resource (like a VM or App Service) that needs to access the Key Vault.
  + Click on "Identity" in the left-hand menu.
  + Turn on the "System assigned" identity and click "Save."

**Step 4:** Access the Secret Programmatically

* **Explanation**: Use the Azure Identity and Key Vault libraries in Python to retrieve the secret.
* **How to Do It**:
  + Install the required libraries:

pip install azure-identity azure-keyvault-secrets

* + Use the following Python code to retrieve the secret:

from azure.identity import DefaultAzureCredential from azure.keyvault.secrets

import SecretClient

credential = DefaultAzureCredential()

kv\_client = SecretClient(vault\_url="https://<key vault name>.vault.azure.net/", credential=credential)

secret = kv\_client.get\_secret("<secret name>")

print(secret.value)

* + Replace **<key vault name>** and **<secret name>** with the actual names of your Key Vault and secret.

**Summary:**

In this lab, you configured Conditional Access to protect your resources, enabled PIM to manage privileged access securely, and used Azure Key Vault for secure storage. These tools enhance your Azure environment's security, ensuring that sensitive data and resources are protected according to best practices.