NetSPI

Modern Adventures in Azure Privilege Escalation

Karl Fosaaen and Thomas Elling

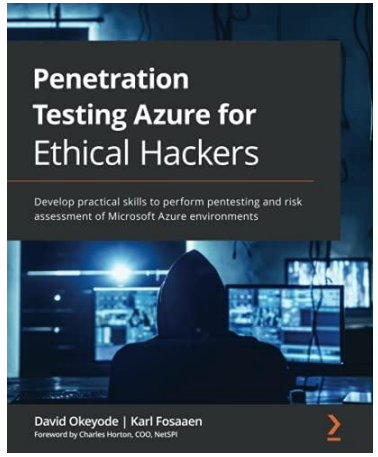
TROOPERS

06.25.26

Get-AzContext

Karl Fosaaen

- VP of Research
- Co-Author - Penetration Testing Azure for Ethical Hackers



FuncoPOP

MicroBurst
TOOLS.NETSPI.COM

Thomas Elling

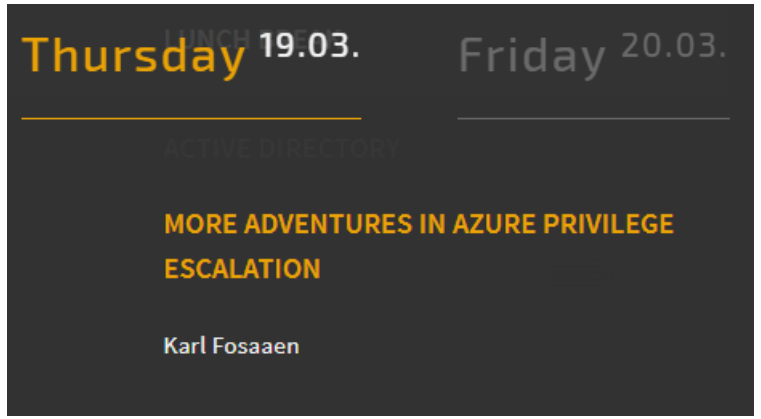
- Director
- Technical Editor - Penetration Testing Azure for Ethical Hackers

Presentation Outline

- Azure Permissions Model
- Azure Escalation Theory
- Initial Access in Azure
- Escalating Read Privileges
- Escalating * Privileges (by Service)
- Pivoting to On-Prem Systems
- Azure Security Wiki
- References / Conclusions

Modern Privilege Escalation?

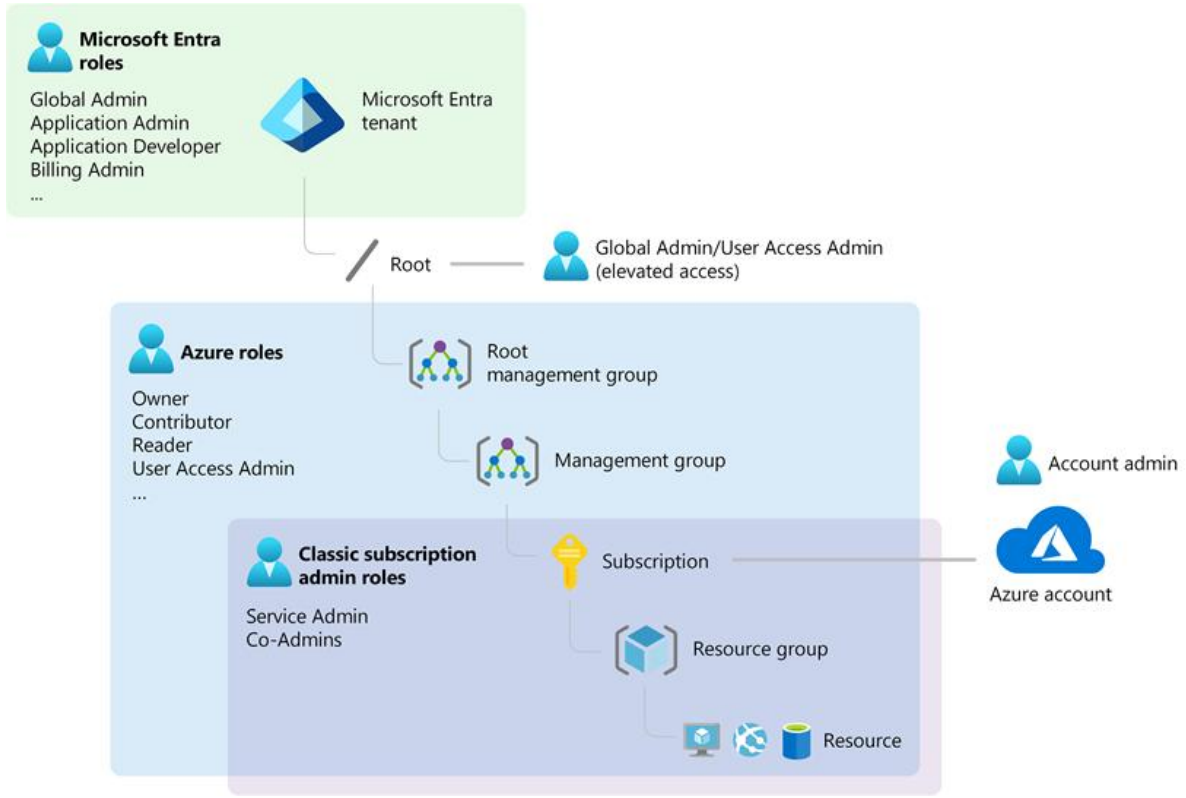
- Azure Cloud has been growing rapidly, including the security attack surface
- As of June 2026: over 200 services/products, 897 Built-in roles, 22,018 different permissions
- New (or updated) tools: AzureHound, ScEntra, Cirro, ROADRecon
- This talk will focus on the techniques behind tooling
  - Azure RBAC concepts
  - Reviewing Azure RBAC from the lens of individual permissions
  - Applying these concepts to example attack chains



Azure Permissions Model

Azure Role Based Access Control (RBAC)

- "who" has access to "what" resources in Azure
- The "who" - user, group, service principal, **managed identity**
- Principal identities are defined in Entra ID -> Azure is inherently linked in this way
- However, Azure RBAC != Entra ID RBAC
- *Almost* zero overlap, with one important exception…



https://learn.microsoft.com/en-us/azure/role-based-access-control/rbac-and-directory-admin-roles

Managed Identities

- Special type of Service Principal where Microsoft completely manages credentials
- Intended for usage with Azure resources when access needs to be granted to those resources
- Have a “servicePrincipalType” of “ManagedIdentity” in Entra ID
- System Assigned vs User Assigned
- “alternativeNames” property contains either the full resource’s resource id for System Assigned or UA resource id
  - “isExplicit=True” set for UA MI

Azure RBAC Crash Course

1 *Resource Hierarchy*

- Management Groups can contain Management Groups + Subscriptions
- Subscriptions can contain Resource Groups
- Resource Groups contain Resources

2 *Scope*

- Level at which a Role Assignment is made
- Role Assignments can be made at each of the Resource Hierarchies
- Roles cascade down

3 *Role Definitions*

- Collections of Permissions defined in JSON
- Built-in and custom roles
- Define allowed and not allowed actions for both control and data planes

4 *Role Assignments*

- Grant access to resources where access is defined by the scope and role definition granted

5 *Permissions*

- Actions that describe an operation on a resource
- Include the Provider, Resource type, Action verb
- *Microsoft.KeyVault/vaults/write*

6 *Control Plane vs Data Plane*

- Control Plane - typically management type actions
- Data Plane - actions related to accessing data in a service

```
{
  "assignableScopes": [
    "/"
  ],
  "description": "Grants full access to manage all resources, including the ability
to assign roles in Azure RBAC.",
  "id": "/providers/Microsoft.Authorization/roleDefinitions/8e3af657-a8ff-443c-
a75c-2fe8c4bcb635",
  "name": "8e3af657-a8ff-443c-a75c-2fe8c4bcb635",
  "permissions": [
    {
      "actions": [
        "*"
      ],
      "notActions": [],
      "dataActions": [],
      "notDataActions": []
    }
  ],
  "roleName": "Owner",
  "roleType": "BuiltInRole",
  "type": "Microsoft.Authorization/roleDefinitions"
}
```

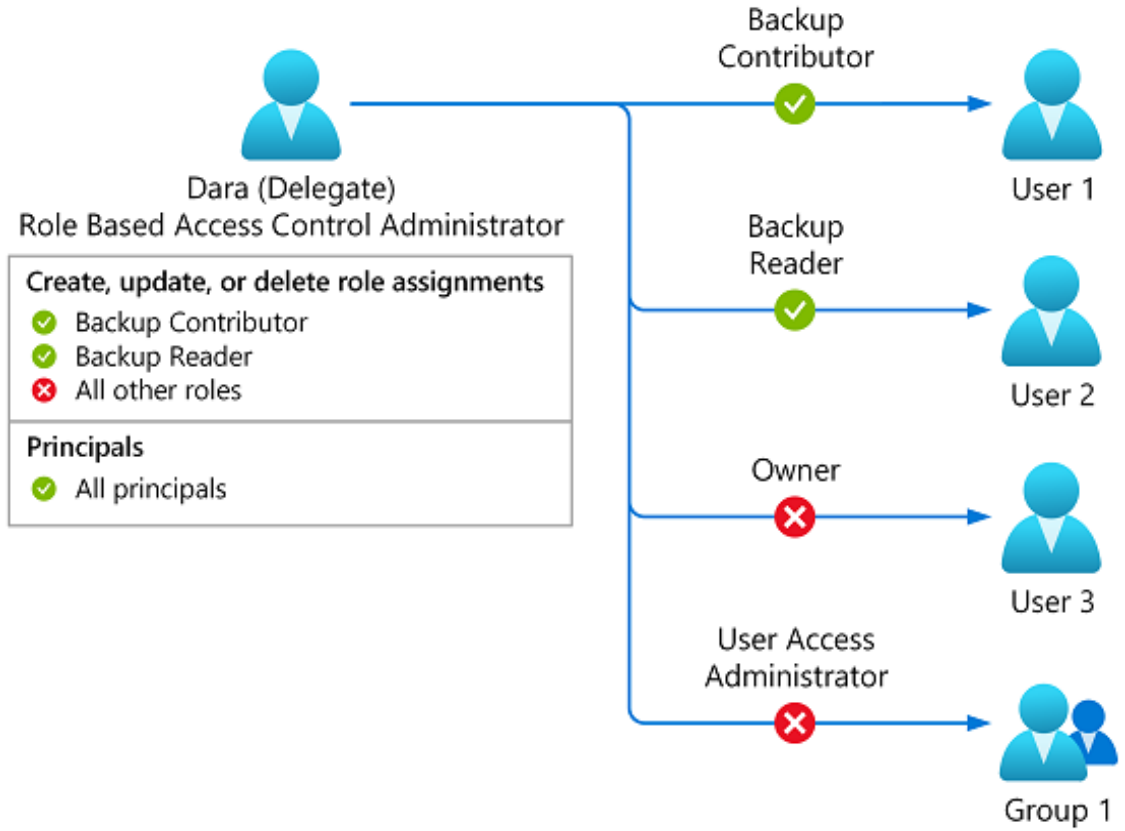

https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/privileged#owner

```
{
  "assignableScopes": [
    "/"
  ],
  "description": "Grants full access to manage all resources, including the ability to assign roles in Azure RBAC.",
  "id": "/providers/Microsoft.Authorization/roleDefinitions/8e3af657-a8ff-443c-a75c-2fe8c4bcb635",
  "name": "8e3af657-a8ff-443c-a75c-2fe8c4bcb635",
  "permissions": [
    {
      "actions": [
        "*"
      ],
      "notActions": [],
      "dataActions": [],
      "notDataActions": []
    }
  ],
  "roleName": "Owner",
  "roleType": "BuiltInRole",
  "type": "Microsoft.Authorization/roleDefinitions"
}
```


https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/privileged#owner

Attribute Based Access Control (ABAC)?

- Fine grained access control model that focuses on attributes
- Can allow for constrained delegation when combined with Azure RBAC
- Often seen in built-in Azure RBAC roles or when attempting to constrain access to Storage Accounts
- Microsoft example shows constrained delegation of roles to other principals where only select roles are allowed
- Conditions use a unique format that is made up of actions, expressions, and attributes



https://learn.microsoft.com/en-us/azure/role-based-access-control/delegate-role-assignments-overview?tabs=template#ways-to-constrain-role-assignments

What

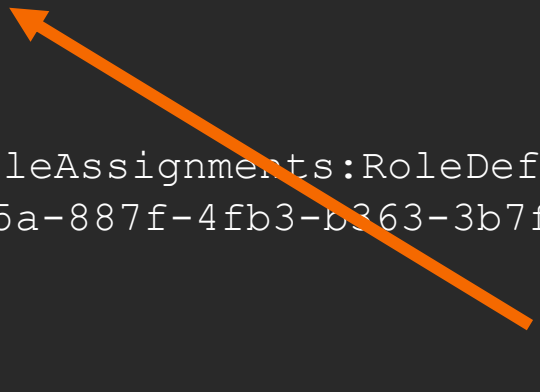
```
((!(ActionMatches{'Microsoft.Authorization/roleAssignments/write'})) OR
(@Request[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483,
a4417e6f-fecd-4de8-b567-7b0420556985, 14b46e9e-c2b7-41b4-b07b-
48a6ebf60603, e147488a-f6f5-4113-8e2d-b22465e65bf6, 12338af0-0e69-4776-
bea7-57ae8d297424, 21090545-7ca7-4776-b22c-e363652d74d2, b86a8fe4-44ce-
4948-aee5-eccb2c155cd7, 4633458b-17de-408a-b874-0445c86b69e6})) AND
((!(ActionMatches{'Microsoft.Authorization/roleAssignments/delete'})) OR
(@Resource[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483,
a4417e6f-fecd-4de8-b567-7b0420556985, 14b46e9e-c2b7-41b4-b07b-
48a6ebf60603, e147488a-f6f5-4113-8e2d-b22465e65bf6, 12338af0-0e69-4776-
bea7-57ae8d297424, 21090545-7ca7-4776-b22c-e363652d74d2, b86a8fe4-44ce-
4948-aee5-eccb2c155cd7, 4633458b-17de-408a-b874-0445c86b69e6}))
```

https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/security#key-vault-data-access-administrator

```
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/write'})
    )
    OR
    (
        @Request[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
AND
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/delete'})
    )
    OR
    (
        @Resource[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
```

Condition

```
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/write'})
    )
    OR
    (
        @Request[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
AND
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/delete'})
    )
    OR
    (
        @Resource[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
```



```
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/write'})
    )
    OR
    (
        @Request[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
AND
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/delete'})
    )
    OR
    (
        @Resource[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
```

Expression

```
(
    (
        !(ActionMatches{'Microsoft.Authorization/roleAssignments/write'})
    )
    OR
    (
        @Request[Microsoft.Authorization/roleAssignments:RoleDefinitionId]
        ForAnyOfAnyValues:GuidEquals{00482a5a-887f-4fb3-b363-3b7fe8e74483, a4417e6f-fecd-
        4de8-b567-7b0420556985, ...}
    )
)
...
```

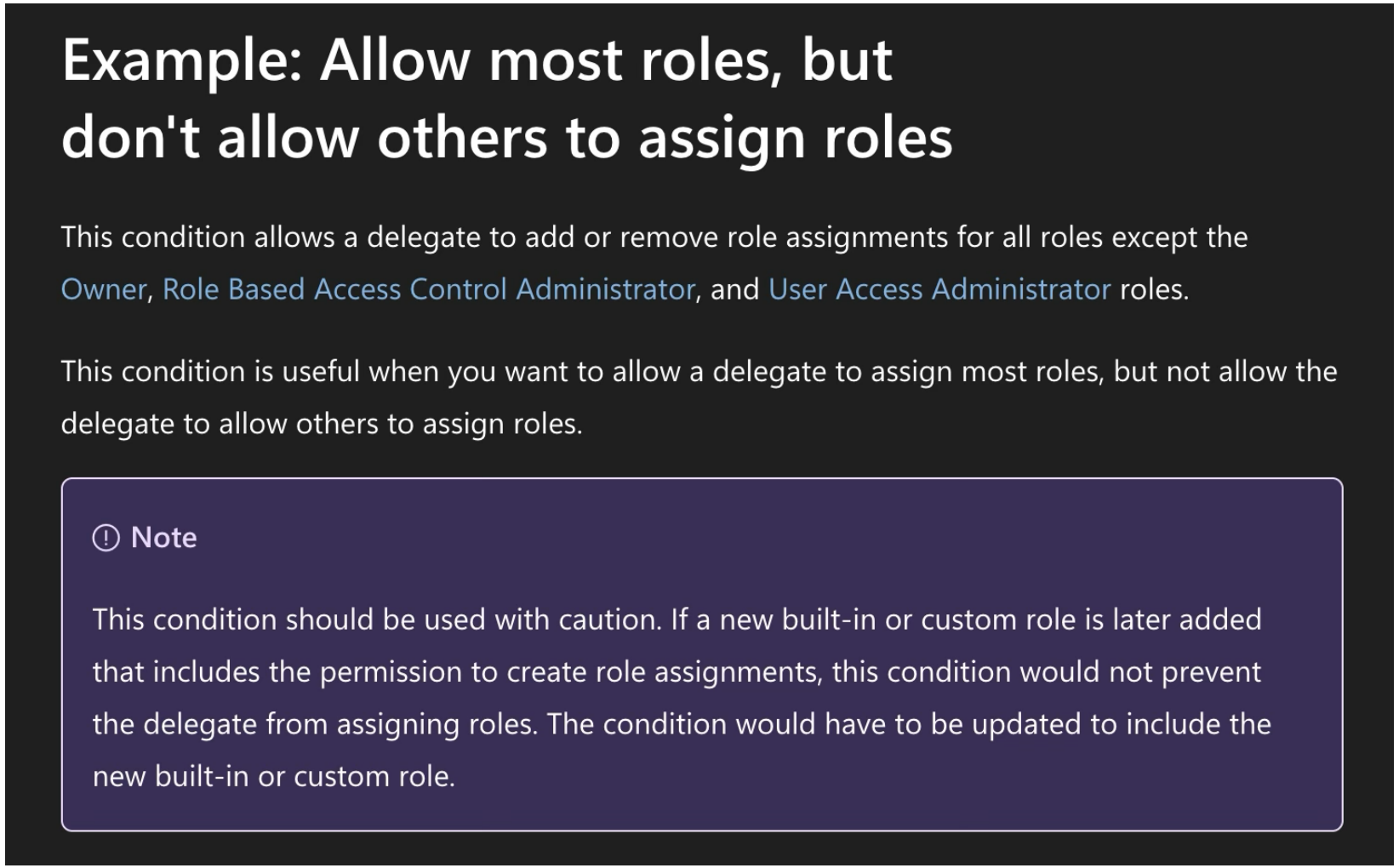
Roughly translates to

1. If the action matches Role Assignments write or delete
2. If the role id equals any of the specified ids in the list, then allow assignment of that role id
3. Else do not allow role assignment

In simpler words

The Key Vault Data Access Administrator role can create or delete role assignments for only the following roles: Key Vault Administrator, Key Vault Certificates Officer, …

Reviewing ABAC Conditions



Example: Allow most roles, but don't allow others to assign roles

This condition allows a delegate to add or remove role assignments for all roles except the Owner, Role Based Access Control Administrator, and User Access Administrator roles.

This condition is useful when you want to allow a delegate to assign most roles, but not allow the delegate to allow others to assign roles.

Note

This condition should be used with caution. If a new built-in or custom role is later added that includes the permission to create role assignments, this condition would not prevent the delegate from assigning roles. The condition would have to be updated to include the new built-in or custom role.

https://learn.microsoft.com/en-us/azure/role-based-access-control/delegate-role-assignments-examples?tabs=template#example-allow-most-roles-but-dont-allow-others-to-assign-roles

A quick note on Graph API Permissions...

- Grants access to the Microsoft Graph API and separate from Entra RBAC roles, but can be extremely privileged

- Comes in two flavors: Delegated vs Application
  - Application permissions (AppRoleAssignments) can be granted directly to Service Principals, including Managed Identities

- Application permissions to know: AppRoleAssignment.ReadWrite.All, RoleManagement.ReadWrite.Directory, Application.ReadWrite.All, etc
  - Merill Fernando has a really cool resource to map permissions to Graph API endpoints - https://graphpermissions.merill.net/permission/

- Not uncommon to see Service Principals/Managed Identities granted Mail.Read in Logic Apps or Automation Accounts

Azure Escalation Theory

Privilege Escalation Categories

- Direct Escalation (RBAC)
  - Ex. *Microsoft.Authorization/roleAssignments/write*
- Execution and Managed Identity Access
  - Ex. *Microsoft.Compute/virtualMachines/runCommand/action*
- User Assigned Managed Identity Access
  - Ex. *Microsoft.ManagedIdentity/userAssignedIdentities/assign/action*
- Secrets Access (Indirect Escalation)
  - Ex. *Microsoft.Storage/storageAccounts/listkeys/action*
- Dependent Service Control
  - Access to a supporting service grants privilege escalation in primary service
  - Ex. *Storage Account Access leads to compromise of Notebooks in Azure Machine Learning services*

Why Focus on Permissions?

- Focusing on permissions can allow for more robust mapping of privilege escalation vectors
  - Built-in AND Custom role coverage
  - Going beyond Reader, Contributor, Owner
  - Privilege escalation vectors in the 897 Built-in roles
  - Some roles have multiple intended functions and cross-service permissions are granted to achieve those functions
- Permissions identified that allow for Direct Escalation, Execution, Managed Identity Abuse across 21 core services
- Also allows for effective privilege escalation mapping when ABAC Conditions are applied
  - ABAC will typically constrain granted permissions, so needs to be considered when applied

Mapping Permissions to Roles

- Can be done offline by gathering all role definitions, review action vs data action, resolve action vs not action etc
- An easier way: an undocumented version of a known ARM API (proxied via interaction with the Azure Portal)
- Allows the caller to filter for Built-in AND custom roles that contain a list of provided permissions
- Handles actions and resolves actions vs not actions
  - However, requires some handling client side due to some odd behavior with data actions
- Given a list of defined ”dangerous” permissions that have been categorized by privilege escalation vector, API will return all roles that contain each set of permissions

API Example

- Role Definitions - List - https://learn.microsoft.com/en-us/rest/api/authorization/role-definitions/list
- Version *2023-07-01-preview* proxied in Azure Portal
- Uses *hasAllPermissions* filter to provide permissions list

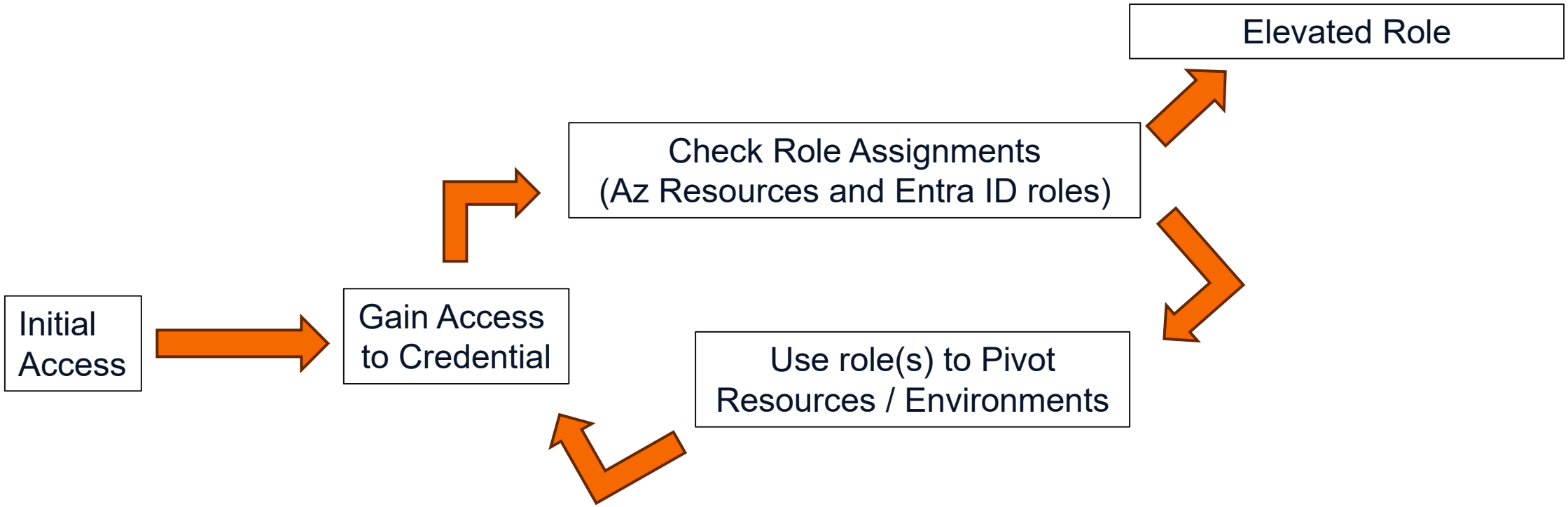
```
> $Path =
"/subscriptions/.../providers/Microsoft.Authorization/roleDefinitions?`$filter=hasAllPermissions('Microsoft.Authorization/elevateAccess/Action')&api-version=2023-07-01-preview"

> $resp = Invoke-AzRestMethod -Method GET -Path $Path

> $resp | Select -ExpandProperty Content | ConvertFrom-Json | Select
@{n="roleName";e={$_.value.properties.roleName}}

roleName
--------
{Owner, User Access Administrator}
```

How do we use this data?

- Ex - *Microsoft.Storage/storageAccounts/listkeys/action*
- “Returns the access keys for the specified storage account.”
  - Control Plane Action, but shared access keys give full access to the Data Plane (read, write, delete)
- Identified in *22 Built-in roles*
- Depending on scope at which roles are granted, could grant unintended access outside of intended resources
- Do you expect roles like Azure AI Administrator, Log Analytics Contributor, or Virtual Machine Contributor to have full data access to Storage Accounts?

```
Owner
Contributor
App Compliance Automation Administrator
Avere Contributor
Azure AI Administrator
Azure Migrate Decide and Plan Expert
Azure Migrate Execute Expert
Azure Migrate Owner
Azure Red Hat OpenShift File Storage Operator
Azure Red Hat OpenShift Image Registry Operator
Azure Red Hat OpenShift Service Operator
DevTest Labs User
Disk Snapshot Contributor
Log Analytics Contributor
Logic App Contributor
Reader and Data Access
SqlMI Migration Role
SqlVM Migration Role
Storage Account Contributor
Storage Account Key Operator Service Role
Virtual Machine Contributor
VM Restore Operator
```

Azure Escalation Theory



Azure Escalation Theory

- Attempting to gain additional access to Subscriptions/Management Groups/etc.
  - Control over the Root Management Group = Control of all Azure Resources
- Attempting to gain full control over the Entra ID Tenant
  - Roles/API Permissions in Entra ID to gain full control over all principals
- Not always a linear path
  - VM Contributor -> Domain Controller VM -> Dump Cloud Admin Creds -> Escalate to Entra Admin

Initial Access in Azure

Initial Access

- **Password Attacks**
  - Password spraying, credential stuffing, etc.
- **Public Credentials**
  - Leaked Keys, Connection Strings, etc.
- **App Vulnerabilities**
  - Command Injection, Data Leakage, SSRF (not likely)
- **Token Gathering**
  - Device code phishing, access token theft
- **Tenant ID Enumeration by Resource**
  - Sending a crafted query to an Azure resource to return the owner Tenant ID

Token Primer

- **Access Tokens:** short-lived JWTs scoped to a resource audience
- **Refresh Tokens:** longer-lived, used to obtain new access tokens
- **PRT (Primary Refresh Token):** device-bound, enables SSO across tenant resources

ROADtools

Dirk-jan's here, go ask him…

What access do we have?

- How to Access/List Your Permissions
  - Azure Portal → Search → Subscriptions
    - Review subscription IAM
  - Azure Portal → Search → Microsoft Entra ID
    - Find your user - “Assigned roles” menu lists your Entra ID roles
  - Az CLI / Az PowerShell / REST API / MicroBurst
  - Azure Resource Manager API - Permissions - List For Resource
    - Gets caller’s permissions at a specific scope and useful when caller may be missing role assignments list permissions
    - https://management.azure.com/subscriptions/{subscriptionId}/providers/Microsoft.Authorization/permissions?api-version=2022-04-01

- Nothing Showing?
  - You have 0 subscriptions, but you may still have the ability to read Entra ID

Escalating Read Privileges

Escalating Read Privileges

- Primary Goals
  - Enumerate resources
  - Find Secrets/Keys/etc.
    - SAS Tokens
  - Reading Configurations
  - Reading Files
- Example Policy →
  - Not a real policy

```
{
…
  "permissions": [
    {
      "actions": [
        "*/read"
      ],
      "notActions": [],
      "dataActions": [
        "Microsoft.Storage/storageAccounts/blobServices/containers
/blobs/read"
      ],
      "notDataActions": []
    }
  ],
 …
}
```

```
{
  "assignableScopes": [
    "/"
  ],
  "description": "View all resources, but does not allow you to make any changes.",
  "id": "/providers/Microsoft.Authorization/roleDefinitions/acdd72a7-3385-48ef-
bd42-f606fba81ae7",
  "name": "acdd72a7-3385-48ef-bd42-f606fba81ae7",
  "permissions": [
    {
      "actions": [
        "*/read"
      ],
      "notActions": [],
      "dataActions": [],
      "notDataActions": []
    }
  ],
  "roleName": "Reader",
  "roleType": "BuiltInRole",
  "type": "Microsoft.Authorization/roleDefinitions"
}
```


https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/general#reader

Reader Role Definition

Reading Resource Configurations

- Control Plane Read Privileges
  - Virtual Machine Extensions Configurations
    - Sensitive data in extension parameters
  - Deployment Templates
    - ARM/Bicep template parameters missing the SecureString flag

Required Permissions

- Microsoft.Compute/virtualMachines/extensions/read
- Microsoft.Resources/deployments/read

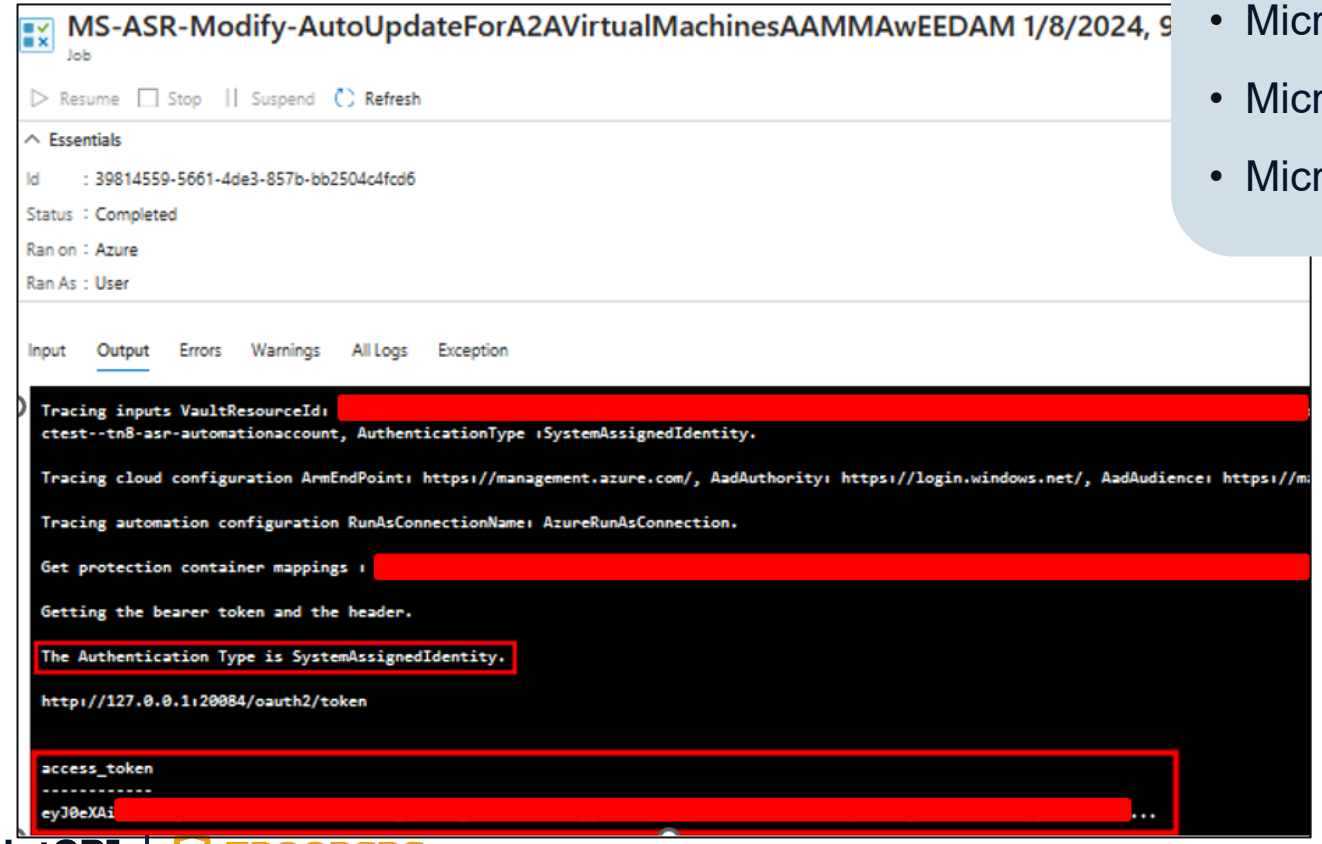
```
Parameters       :
                 Name             Type             Value
                 ===============  ===============  ==========
                 location         String           centralus
                 vmssName         String           testscale
                 vmSku            String           Standard_B1s
                 adminUsername    String           AZAdmin
                 instanceCount    String           2
           [Truncated]
                 publicIpAddressPerInstance  String  false
                 upgradeMode      String           Manual
                 adminPassword    String           IsThisCleartext?
```

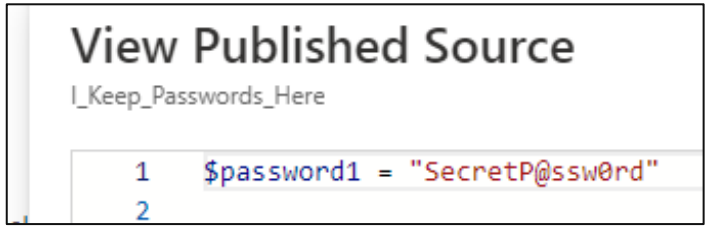
Reading Resource Configurations

- Control Plane Read Privileges
  - Automation Accounts
    - Runbook Code and Parameters
    - Sensitive Data in Job Outputs

Required Permissions

- Microsoft.Automation/automationAccounts/read
- Microsoft.Automation/automationAccounts/jobs/read
- Microsoft.Automation/automationAccounts/jobs/output/read
- Microsoft.Automation/automationAccounts/jobs/streams/read
- Microsoft.Automation/automationAccounts/runbooks/read





Elevating Privileges with Azure Site Recovery Services

March 28, 2024

Joshua Murrell

Reading Resource Configurations

- Control Plane Read Privileges
  - Azure Container Registries
    - Read permissions can “Pull” container images
    - Run the container image and review
      - ENV Variables
      - Local files

Required Permissions

- Microsoft.ContainerRegistry/registries/pull/read

Helpful Permissions

- Microsoft.ContainerRegistry/registries/read

```
/ # printenv
HOSTNAME=288bc051cffc
ServicePrincipalSecret=u[REDACTED]3
SHLVL=1
HOME=/root
ServicePrincipalAppID=0eaefb61-6a09-4ed4-9305-f119d29d9224
TERM=xterm
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
ServicePrincipalTenantID=72f988bf-86f1-41af-91ab-2d7cd011db47
PWD=/
```

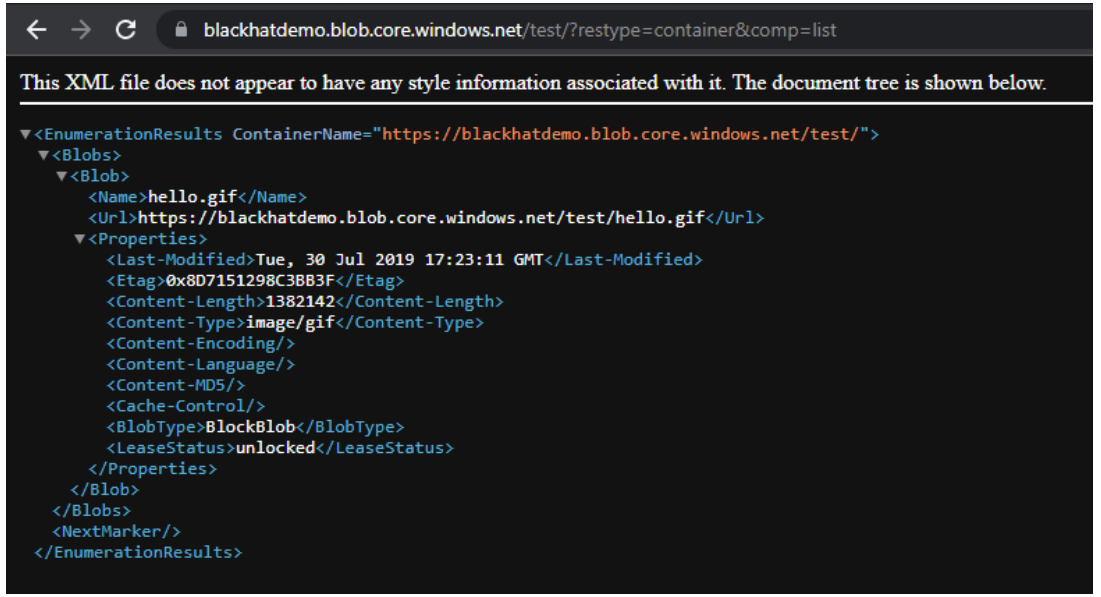
```
{
  "assignableScopes": [
    "/"
  ],
  "description": "Allows for read access to Azure Storage blob containers and data",
  "id": "/providers/Microsoft.Authorization/roleDefinitions/2a2b9908-6ea1-4ae2-8e65-
a410df84e7d1",
  "name": "2a2b9908-6ea1-4ae2-8e65-a410df84e7d1",
  "permissions": [
    {
      "actions": [
        "Microsoft.Storage/storageAccounts/blobServices/containers/read",
        "Microsoft.Storage/storageAccounts/blobServices/generateUserDelegationKey/action”
      ],
      "notActions": [],
      "dataActions": [
        "Microsoft.Storage/storageAccounts/blobServices/containers/blobs/read"
      ],
      "notDataActions": []
    }
  ],
  "roleName": "Storage Blob Data Reader",
  "roleType": "BuiltInRole",
  "type": "Microsoft.Authorization/roleDefinitions"
}
```

https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/storage#storage-blob-data-reader

Storage Blob Data Reader

Reading Resource Configurations

- Data Plane Read Privileges
  - Storage Accounts
    - Microsoft.Storage/storageAccounts/blobServices/containers/blobs/read
      - Best paired with Control Plane read action to list containers
        - Microsoft.Storage/storageAccounts/blobServices/containers/read
    - Storage Account “Container” with “Container” anonymous access permissions
      - Configuration read permissions allow you to see the permissions
      - Container permissions allow you to list the files
        - *?restype=container&comp=list*



Escalating Read Privileges

- List of additional services we don’t have time for
  - Container Instances
  - Container Apps
  - Logic Apps
  - Data Factory
  - Custom Connections
  - Batch

Escalating * Privileges (by Service)

Escalating “*“ Privileges

- Primary Goals
  - Find Secrets/Keys/etc.
  - Command execution
  - Reading Configurations
  - Reading Files
- Example Policy →
  - Not a real policy
- “*”, Action, and/or Write
  - Control Plane
  - Data Plane

```
{
…
  "permissions": [
    {
      "actions": [
        "Microsoft.Compute/virtualMachines/extensions/*",
        "Microsoft.Storage/storageAccounts/listKeys/action"
      ],
      "notActions": [],
      "dataActions": [
       "Microsoft.Storage/storageAccounts/blobServices/containers
/blobs/write"
        ],
      "notDataActions": []
    }
  ],
 …
}
```

Escalating “*“ on Virtual Machines

- Command Execution as a Service
  - Run Command API
  - Run Command Deployment
  - Custom Script Extensions
  - DSC Extension
  - Serial Console
- All run as the System/Root account

- Commands to Run
  - “Totally Normal Windows Stuff…”
    - Looking at files, dumping LSASS…

- Domain controllers are often in the cloud…

Required Permissions

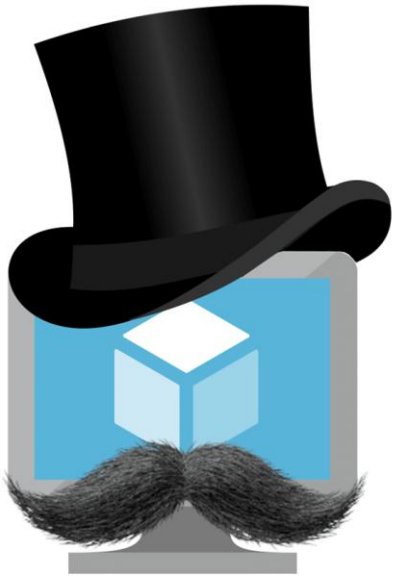
- Microsoft.Compute/virtualMachines/extensions/write
- Microsoft.Compute/virtualMachines/runCommand/action
- Microsoft.Compute/virtualMachines/runCommands/write
- Microsoft.Compute/virtualMachineScaleSets/extensions/write
- Microsoft.Compute/virtualMachineScaleSets/virtualMachines/runCommand/action
- Microsoft.Compute/virtualMachineScaleSets/virtualMachines/runCommands/write
- Microsoft.SerialConsole/serialPorts/connect/action

Helpful Permissions

- Microsoft.Compute/virtualMachines/read

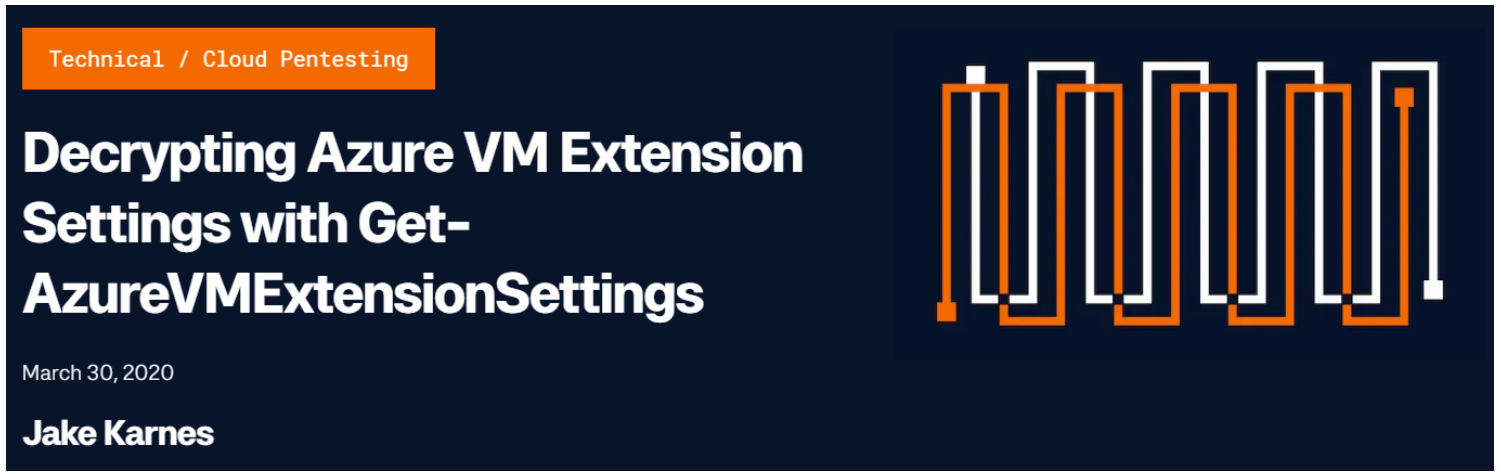
Escalating “*“ on Virtual Machines

- Escalating on/from the VM
  - Instance Metadata Service (IMDS) - 169.254.169.254
    - Generate Managed Identity Tokens
      - http://169.254.169.254/metadata/identity/oauth2/token?...[Truncated]
  - WireServer Service - 168.63.129.16
    - Minimally documented orchestration harness for VM deployment
    - Only routable from Azure VMs as a local admin
    - Can contain encrypted configs from VM Extensions
      - We can decrypt those…



Escalating “*“ on Virtual Machines

- Escalating on/from the VM
  - Review of VM Extension Files
    - Different from the configurations we looked at with read permissions
    - Configs can contain encrypted sensitive information
      - Example: C:\Packages\Plugins\Microsoft.CPlat.Core.RunCommandWindows\1.1.15\Downloads
      - We can also decrypt these…
    - Requires Local Admin



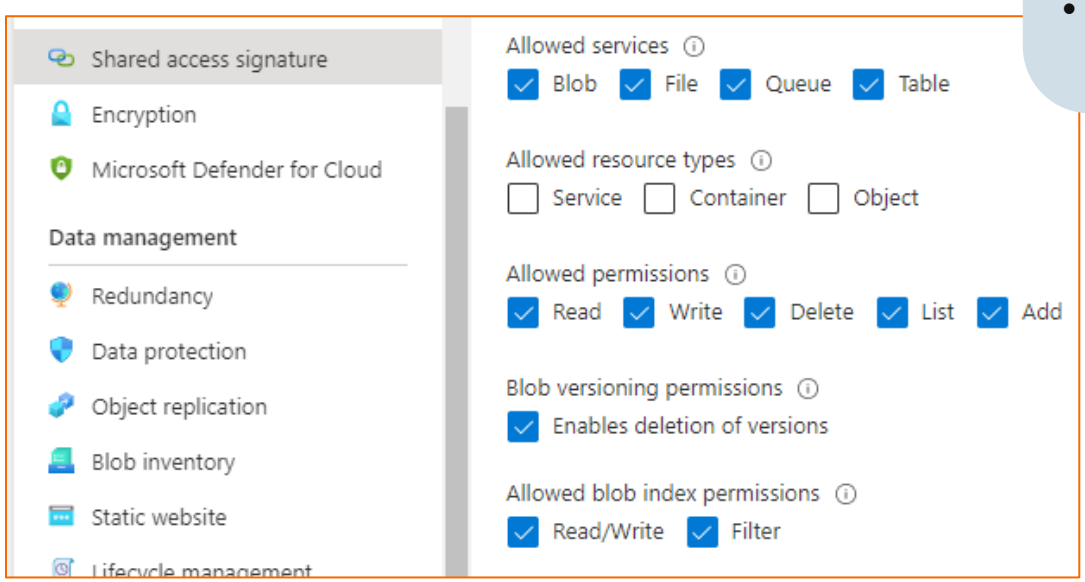
Escalating “*“ on Virtual Machines

- Log Analytics Contributor Role issues
  - Meant as more of a logs management role, but allows VM code execution...

```
...
"actions": [
  "*/read",
  "Microsoft.ClassicCompute/virtualMachines/extensions/*",
  "Microsoft.ClassicStorage/storageAccounts/listKeys/action",
  "Microsoft.Compute/virtualMachines/extensions/*",
  "Microsoft.HybridCompute/machines/extensions/write",
  "Microsoft.Insights/alertRules/*",
  "Microsoft.Insights/diagnosticSettings/*",
  "Microsoft.OperationalInsights/*",
  "Microsoft.OperationsManagement/*",
  "Microsoft.Resources/deployments/*",
  "Microsoft.Resources/subscriptions/resourcegroups/deployments/*",
  "Microsoft.Storage/storageAccounts/listKeys/action",
  "Microsoft.Support/*"
],
...
```

Escalating “*“ on Storage Accounts

- **Sensitive Data Access**
  - List Files, Read Files, …, Profit?
- **Managing Access**
  - Key-based Access
  - Permission-based Access
  - SAS Tokens



Required Data Plane Permissions

- Microsoft.Storage/storageAccounts/blobServices/containers/blobs/write
- Microsoft.Storage/storageAccounts/fileServices/fileshares/files/write

Required Control Plane Permissions

- Microsoft.Storage/storageAccounts/listKeys/action

Escalating “*“ on Storage Accounts

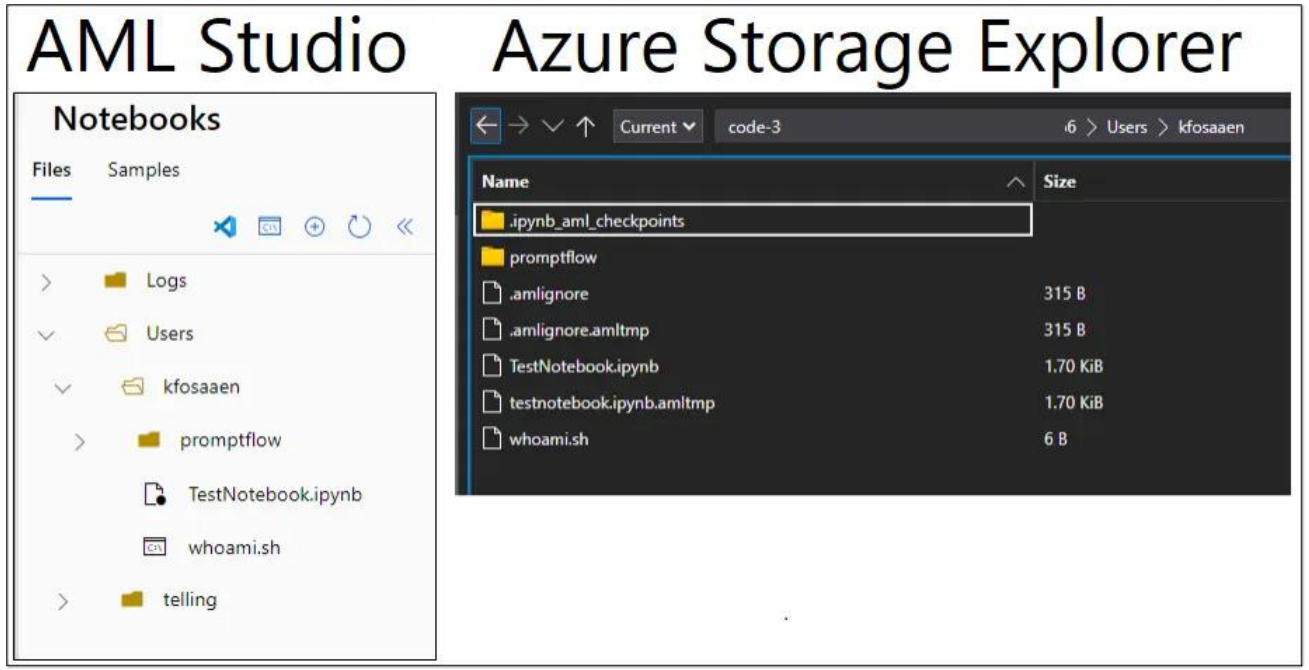
- **Cross-Service Command Execution**
  - Function Apps / Logic Apps (technically the same)
    - FuncoPop Tool
  - Cloud Shell
  - Azure Batch
  - Azure Data Factory
  - Azure Machine Learning

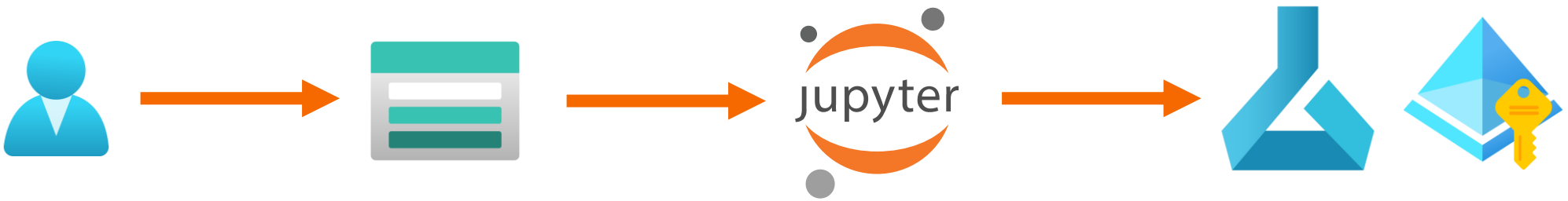
Required Permissions

- Microsoft.Storage/storageAccounts/listkeys/action
- Microsoft.Storage/storageAccounts/listAccountSas/action
- Microsoft.Storage/storageAccounts/listServiceSas/action

Dependent Service Control

- Azure Machine Learning stores user created Jupyter notebooks in Storage Account File Shares
- Write access to the file shares -> can modify notebooks with no permissions on the actual AML service
- Code execution and managed identity control



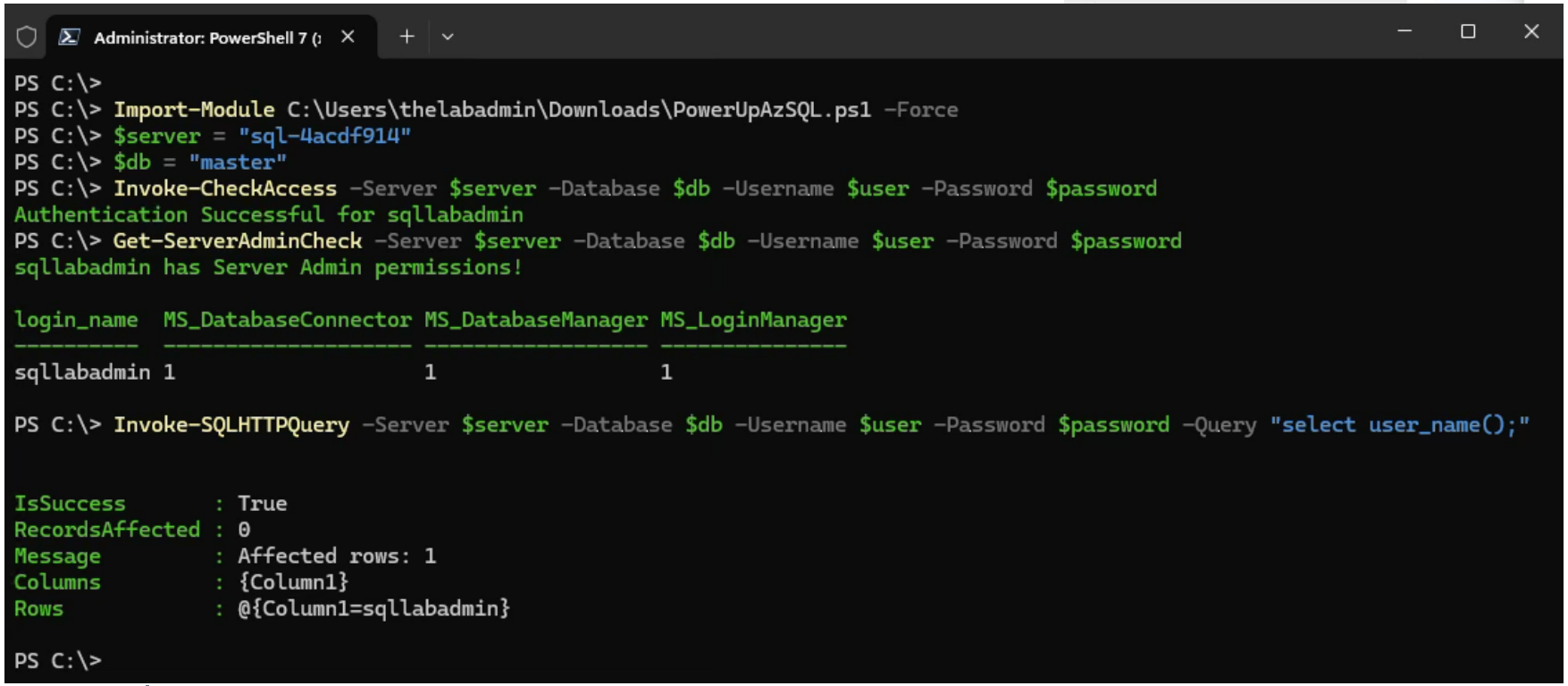


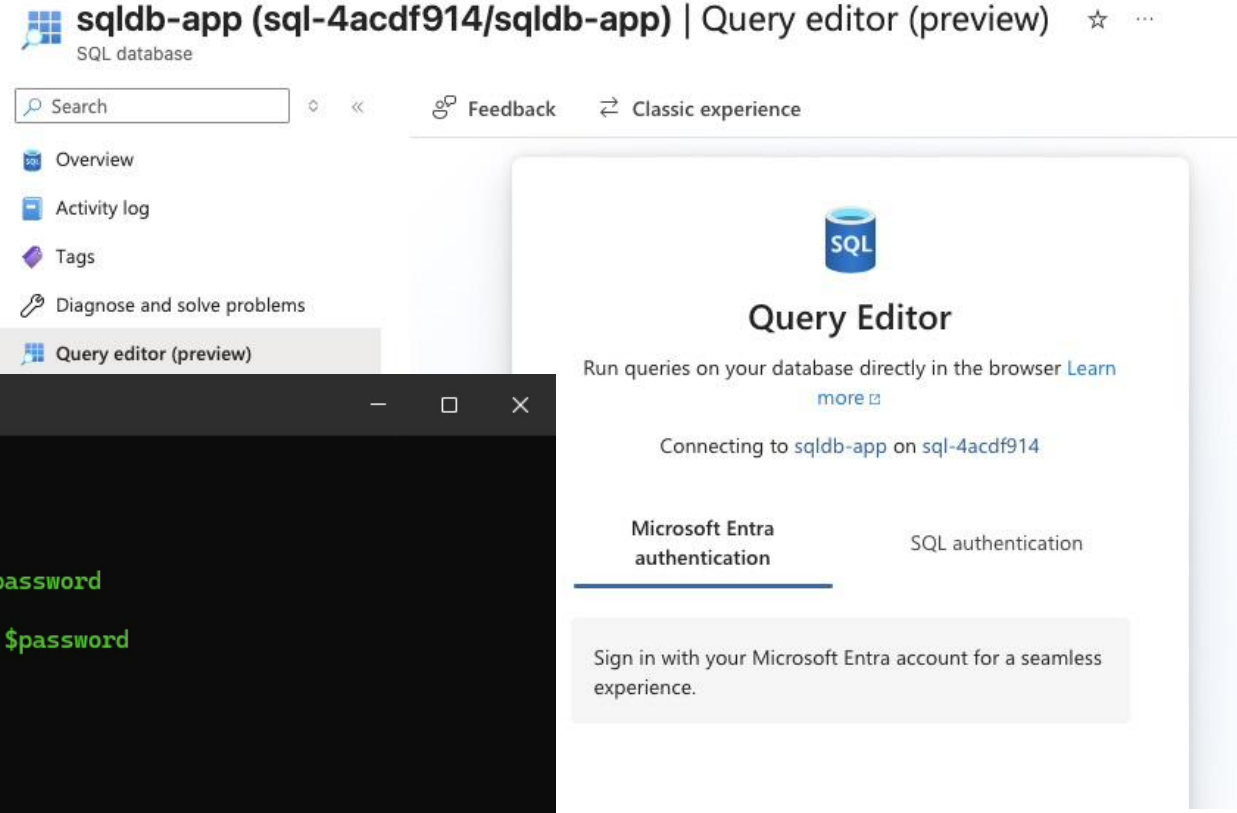
Everything Else

- **We don’t have time for these, but these all support Managed Identities (and allow for token extraction)**
  - Automation Accounts
  - Azure Machine Learning
  - Deployment Scripts
  - Container Registries
  - Azure Kubernetes Service
  - App Services
  - Azure Batch
  - Logic Apps

Escalating – Azure SQL Database

- **Unexpected Control Channels**
  - HTTP access to Azure SQL Database service
    - Credential brute forcing
    - SQL query execution
    - Entra ID authentication





Escalating “*“ Privileges Everywhere

- Assume you have * at the Subscription level (Owner and Contributor roles)
  - Dumping Secrets/Keys/etc. at Scale
  - Get-AzPasswords in MicroBurst
    - Covers the following services:
      - Key Vaults*
      - Azure Kubernetes Service (AKS)
      - Azure Container Registry
      - App Services
      - Function Apps
      - Automation Accounts
      - StorageAccounts
      - CosmosDB
      - ContainerApps
      - API Management (APIM)
      - ServiceBus
      - AppConfiguration
      - Azure Batch
      - Cognitive Services

* Allows you to grant Access Policy to yourself – Until they take those away in 2027

Pivoting to On-Prem Systems

Pivoting to On-Prem Systems

- **ExpressRoute**
  - Connections to on-premises
    - Pivot from VMs
- **ARC**
  - Command execution
    - Run Command
    - VM Extensions
- **Intune**
  - Command execution

Required Permissions

- Microsoft.Compute/virtualMachines/extensions/write
- Microsoft.Compute/virtualMachines/runCommand/action
- Microsoft.Compute/virtualMachines/runCommands/write
- Microsoft.Compute/virtualMachineScaleSets/extensions/write
- Microsoft.Compute/virtualMachineScaleSets/virtualMachines/runCommand/action
- Microsoft.Compute/virtualMachineScaleSets/virtualMachines/runCommands/write
- Microsoft.SerialConsole/serialPorts/connect/action
- Microsoft.HybridCompute/machines/write

Helpful Permissions

- Microsoft.Compute/virtualMachines/read

Sample Escalation Path

- **Initial Access**
  - Command injection in App Services application
- **Credential Access**
  - Application has Service Principal in ENV variables
  - Service Principal has additional permissions
- **SP Can Read ACR images**
  - Additional credentials out of the image
- **Additional creds have actions available**
  - Command Exec, etc.
- **Permissions to assign roles**
  - AppRoleAssignment.ReadWrite.All
  - Loosely follow the midnight blizzard escalation

Persistence

Persisting in Azure

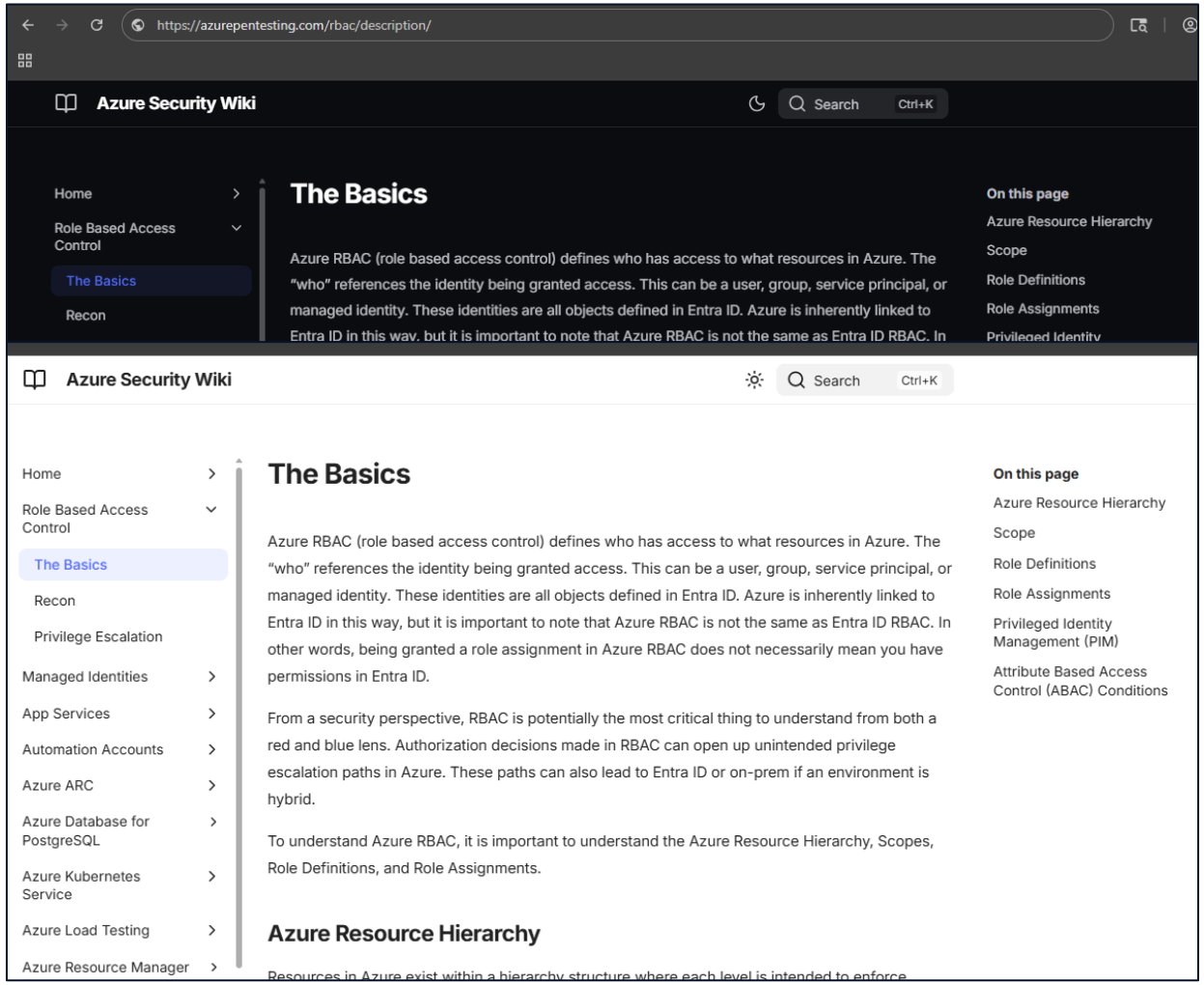
- **Entra ID Level**
  - Adding Accounts
  - Adding App Registrations
  - Adding Credentials to Existing Apps
    - Secrets / Certificates / Federated Credentials
  - Adding Federated Credentials to User-Assigned Managed Identities
  - Adding Entra roles to principals

- **Resource Level**
  - Adding Resource Credentials – ACR Admin, VM Local Admins, Azure SQL local users, etc.
  - Backdoor existing code bases – Automation Runbooks, AML Notebooks, Function Apps, etc.
  - Adding Subscription roles to principals
  - Exporting Managed Identity Certificates (API Management and ARC)

Azure Security Wiki

Azure Security Wiki

- **Purpose**
  - Baseline RBAC/permissions overview
  - Attack primitives by service
    - Required permissions
    - MITRE mappings
- **Open Source and Open to Contributions**
  - https://azurepentesting.com/
  - GitHub
    - https://github.com/NetSPI/AzurePentestingWiki
  - Contributing Guidelines
    - Try to write the content yourself



References / Conclusions

Conclusions

- The core fundamentals for securing RBAC still apply in modern cloud environments
- Recommendations
  - Logical segmentation of workloads
  - Review every role (even Built-in roles) for unexpected permissions before assignment
  - Assign the least privileged role possible at the most narrow scope possible
- More important than ever when AI can accelerate capabilities

Thanks!

- **Previous Research**
  - In no particular order:
    - Dirk-jan Mollema, Dr Nestori Syynimaa, Josh Magri, Jake Karnes, Katie Knowles, Eric Woodruff, Thomas Naunheim, Leron Gray, Andy Robbins, Daniel Heinsen, Ryan Hausknecht, NyxGeek, Beau Bullock, David Okeyode, Rogier Dijkman, Cody Burkard, Raunak Parmar, Chirag Savla, Clément Notin, Nathan McNulty, Kat Traxler, Andy Gill, Karim El-Melhaoui, Kinnaird McQuade, Simon Maxwell-Stewart, Christian Bortone, Nitesh Surana, Christian Philipov, Seth Art, Aled Mehta, Merill Fernando, Sapir Federovsky, Shahar Dorfman
- **Everyone at NetSPI**
- **Thank YOU all for attending the talk!**

Say hello!

Karl Fosaaen

- Technical Blog – https://www.netspi.com/authors/karl-fosaaen
- Everything else social media – @kfosaaen
- LinkedIn – Karl-Fosaaen

Thomas Elling

- Technical Blog – https://www.netspi.com/authors/telling
- LinkedIn – thomaselling1