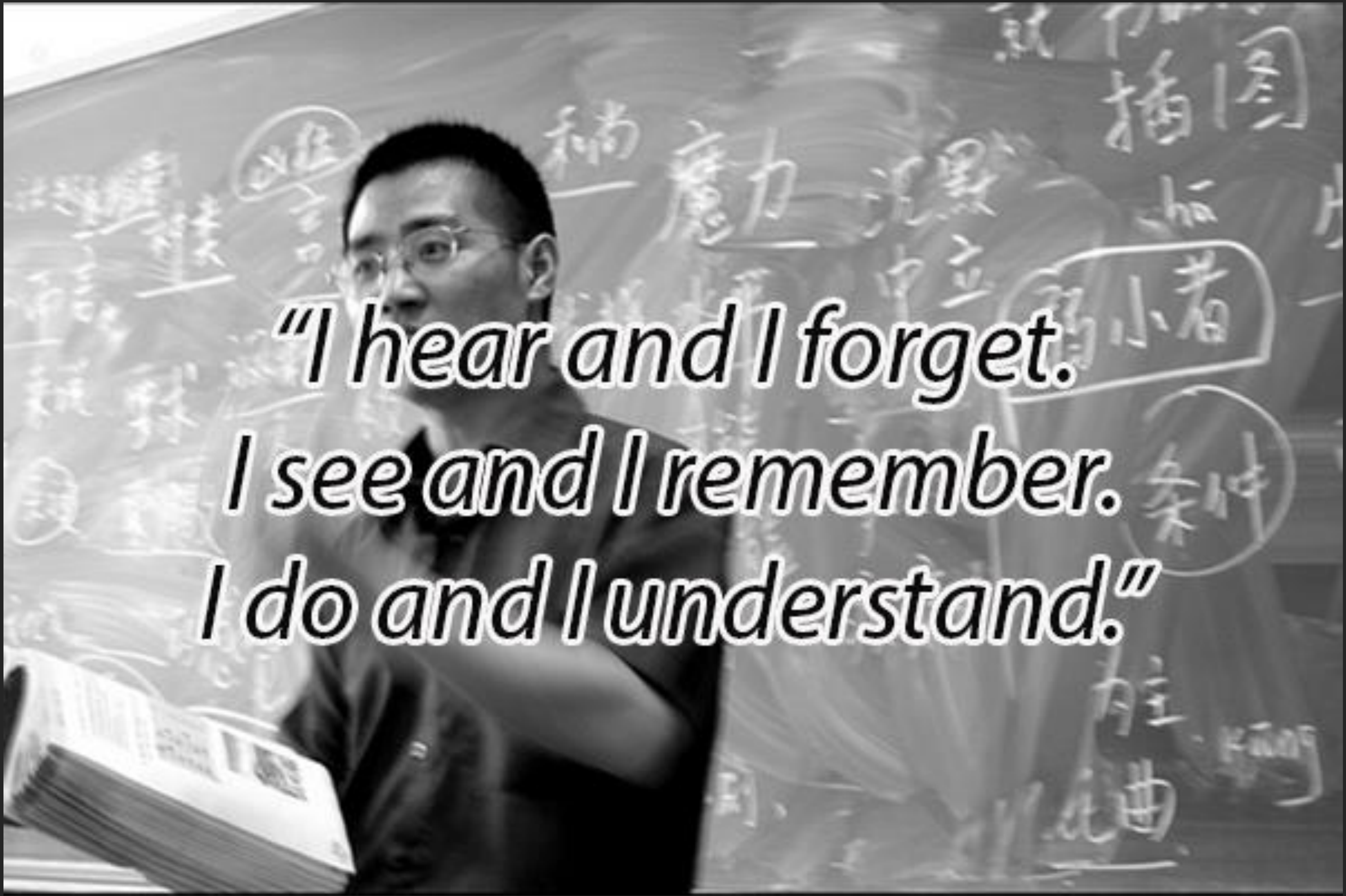
#NVSummit2021

Protecting Azure Serverless Solutions with Azure AD

- Jan Vidar Elven
- Senior Architect @ Skill AS
- @JanVidarElven
- gotoguy.blog
- MVP Enterprise Mobility

NORDIC – VIRTUAL SUMMIT –



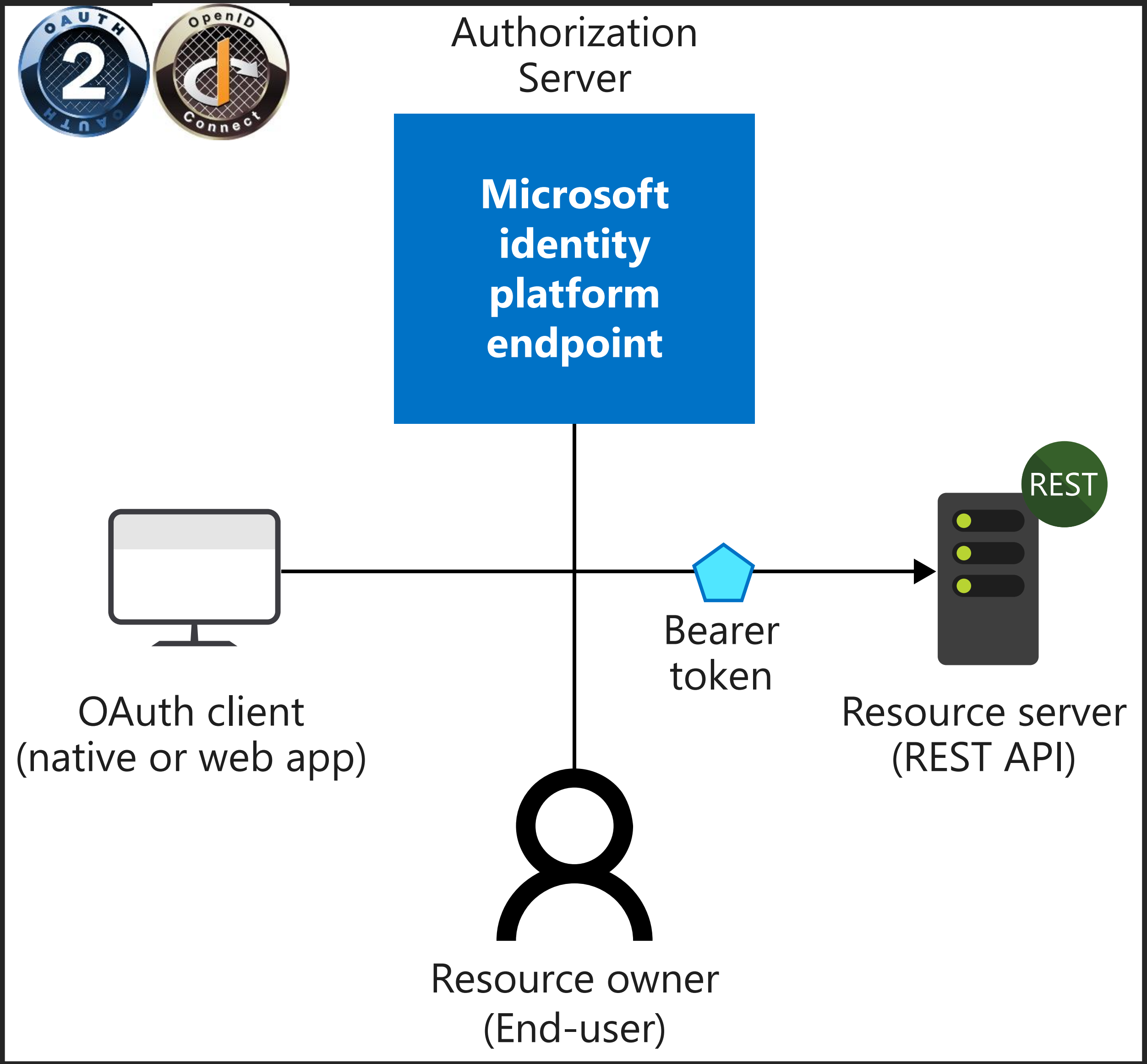
> “I hear and I forget.
> I see and I remember.
> I do and I understand.”

Marco Klapper under CC-BY-SA licence. Text added on top by The Open University. under Creative-Commons license

Lets get the basics straight..

Using Microsoft Identity Platform for Authentication and Authorization

Microsoft Identity, OAuth2 & Open ID Connect


OAUTH 2 OAUTH
OpenID Connect
Authorization Server
Microsoft identity platform endpoint
REST
OAuth client (native or web app)
Bearer token
Resource server (REST API)
Resource owner (End-user)

Why Protect ServerLess Solutions?

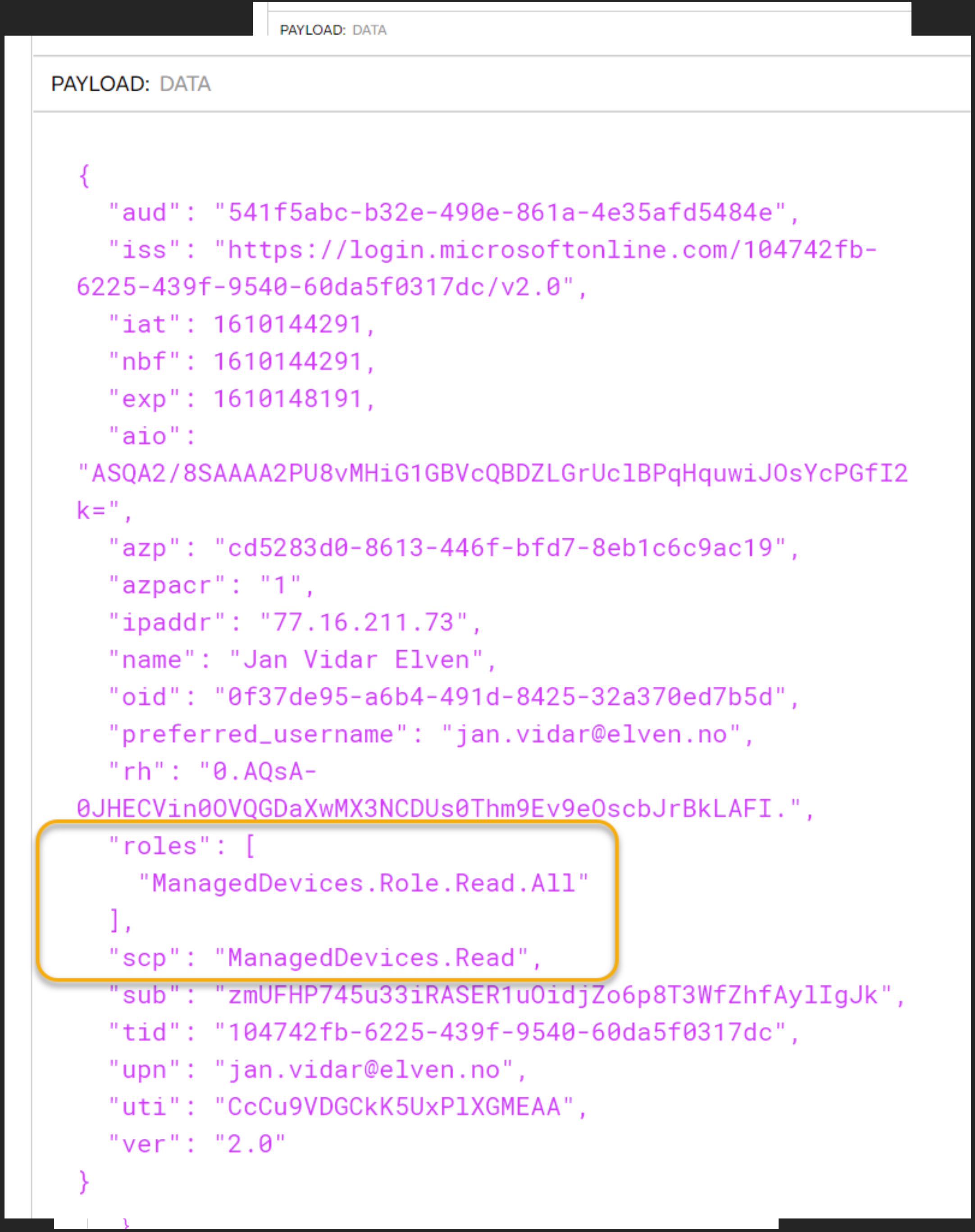
- Serverless Solutions use Triggers
  - Events, Timer, Queue etc
  - Connectors with Trigger actions in Logic Apps Workflows
- HTTP Request Triggers for manual invoke
  - Serverless can be API (REST)
  - Default protection with API key, SAS (Shared Access Signature), IP restrictions etc
- Azure AD Oauth2 provides authentication and authorization
  - Can use the Powers of Azure AD! (MFA, Conditional Access, Risk, etc)

The Token!

- OAuth 2.0
- OpenID Connect
  - Identity verification
  - ID Token
- JSON web tokens (JWT) Base64 encoded
- ID Token vs Access Token
- Access Token
  - For authorization and access to API
  - Access Token audience configured via App Registration in Azure AD

API Permissions

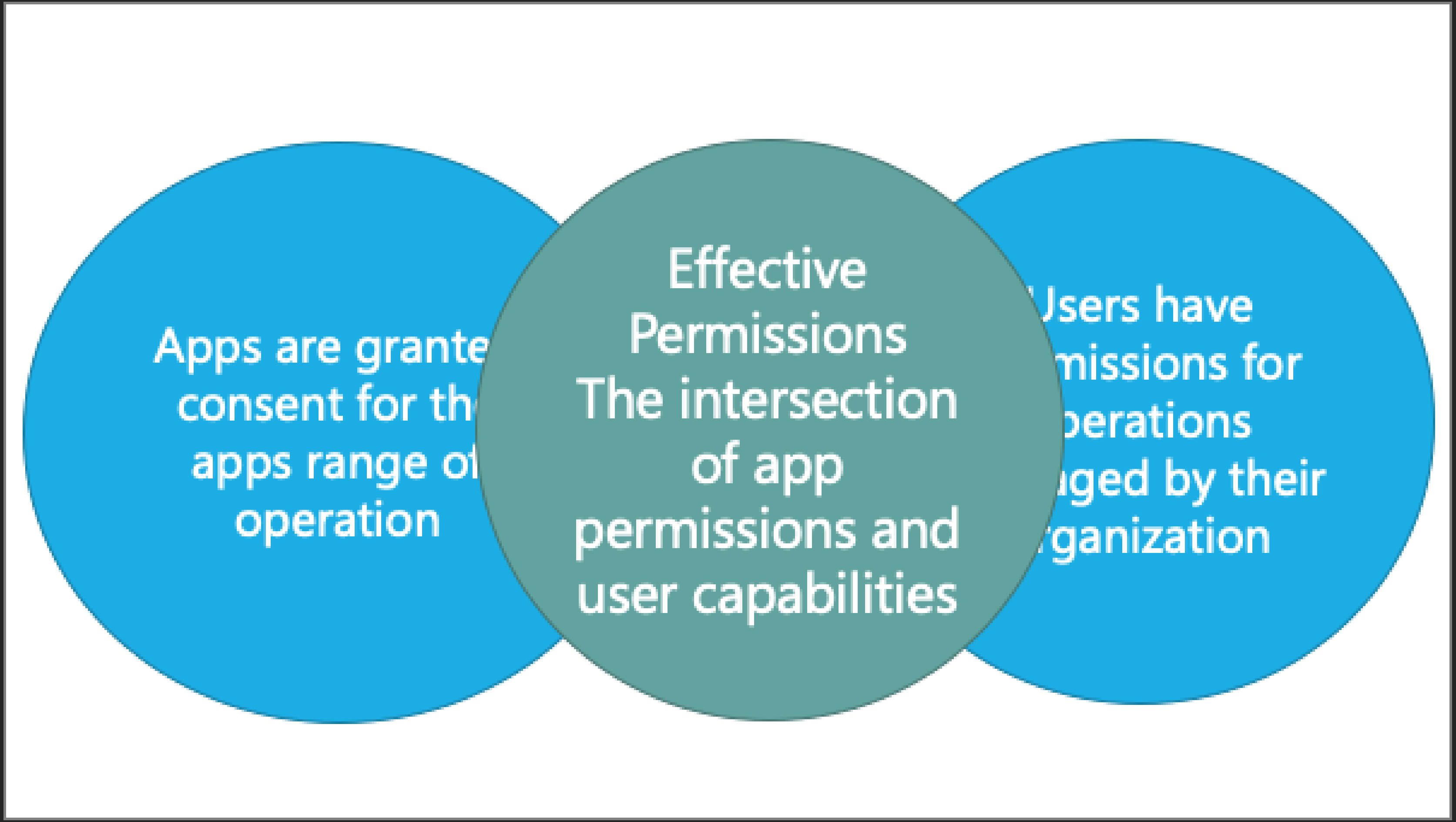
- Delegated Permissions
  - On behalf of user
  - Scopes
    - String (Comma separated)
  - User/Admin consent
- Application Permissions
  - As application/deamon
  - Roles
    - Array
  - Admin consent



PAYLOAD: DATA

```
{
  "aud": "541f5abc-b32e-490e-861a-4e35afd5484e",
  "iss": "https://login.microsoftonline.com/104742fb-
6225-439f-9540-60da5f0317dc/v2.0",
  "iat": 1610144291,
  "nbf": 1610144291,
  "exp": 1610148191,
  "aio":
"ASQA2/8SAAAA2PU8vMHiG1GBVcQBDZLGrUclBPqHquwiJOsYcPGfI2
k=",
  "azp": "cd5283d0-8613-446f-bfd7-8eb1c6c9ac19",
  "azpacr": "1",
  "ipaddr": "77.16.211.73",
  "name": "Jan Vidar Elven",
  "oid": "0f37de95-a6b4-491d-8425-32a370ed7b5d",
  "preferred_username": "jan.vidar@elven.no",
  "rh": "0.AQsA-
0JHECVin0OVQGDaXwMX3NCDUs0Thm9Ev9eOscbJrBkLAFI.",
  "roles": [
    "ManagedDevices.Role.Read.All"
  ],
  "scp": "ManagedDevices.Read",
  "sub": "zmUFHP745u33iRASER1uOidjZo6p8T3WfZhfAylIgJk",
  "tid": "104742fb-6225-439f-9540-60da5f0317dc",
  "upn": "jan.vidar@elven.no",
  "uti": "CcCu9VDGCkK5UxPlXGMEAA",
  "ver": "2.0"
}
```

Delegated vs Application Permissions



Apps are grante
consent for th
apps range of
operation

Effective
Permissions
The intersection
of app
permissions and
user capabilities

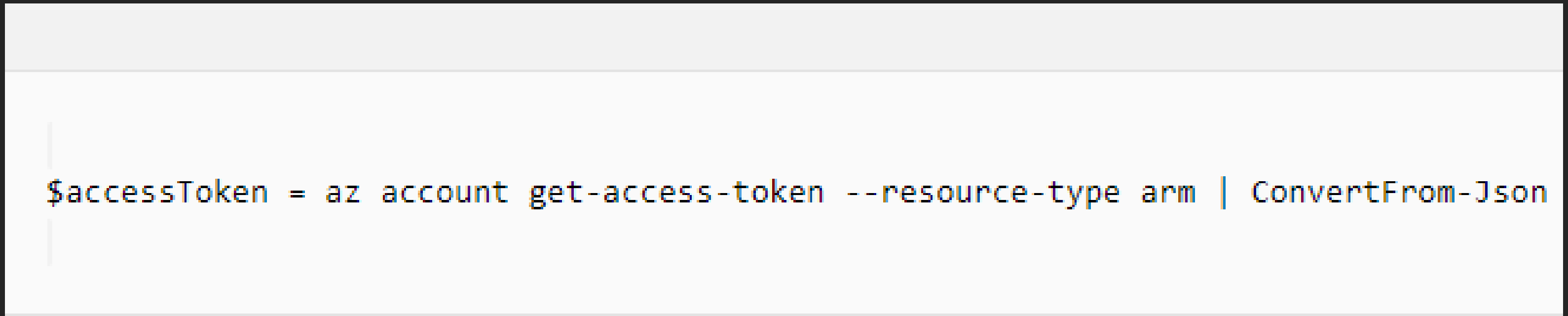
Users have
missions for
perations
ged by their
rganization

Scenario – Management Access

Requesting Tokens from well known Management Clients and Use for Authorization

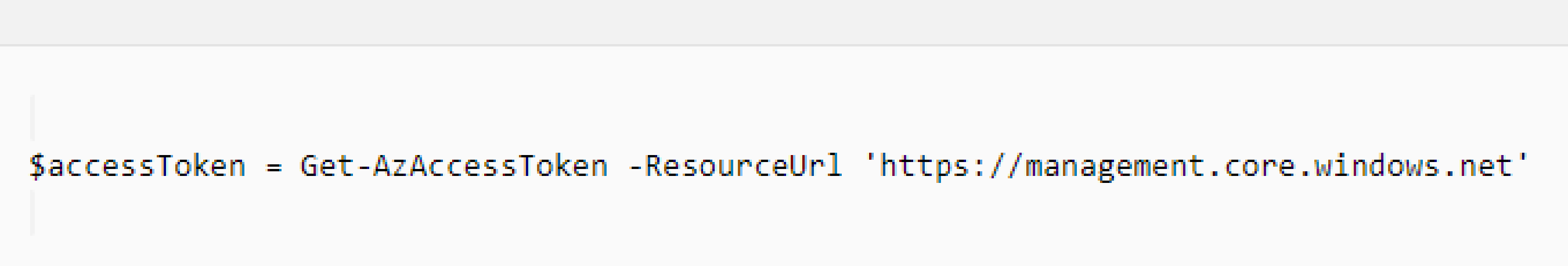
Getting an Management Access Token

- Azure CLI



```
$accessToken = az account get-access-token --resource-type arm | ConvertFrom-Json
```

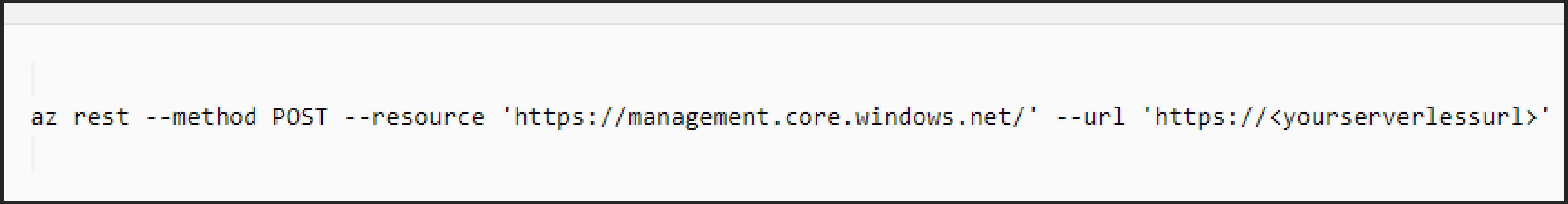
- Az PowerShell



```
$accessToken = Get-AzAccessToken -ResourceUrl 'https://management.core.windows.net'
```

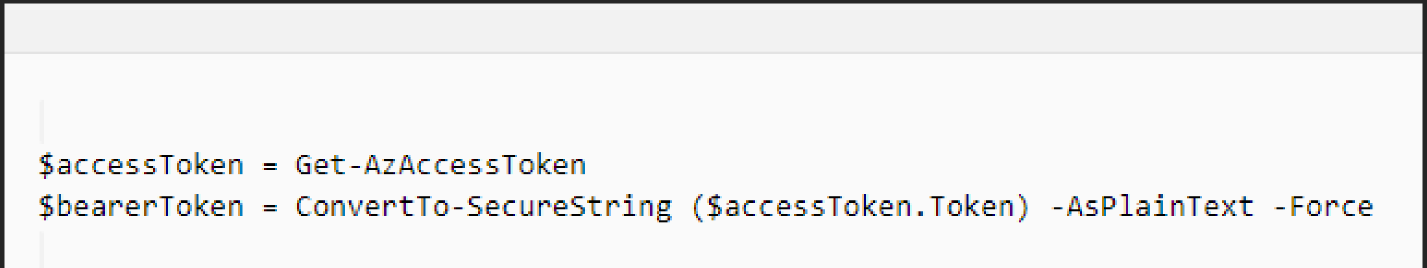
Sending Authorized Requests

- Azure CLI

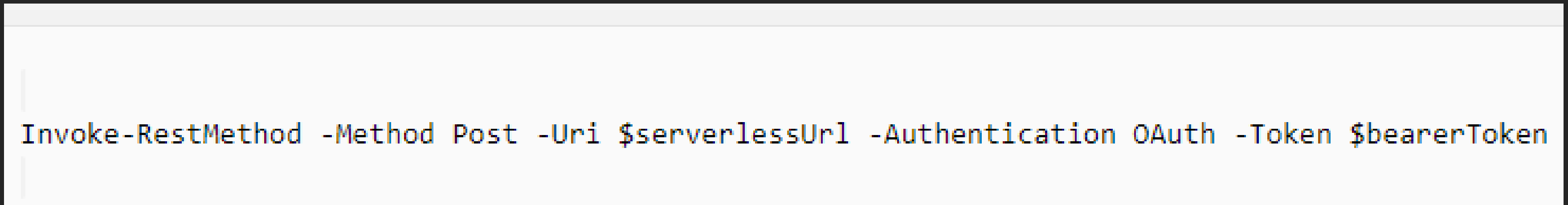


```
az rest --method POST --resource 'https://management.core.windows.net/' --url 'https://<yourserverlessurl>'
```

- Azure PowerShell



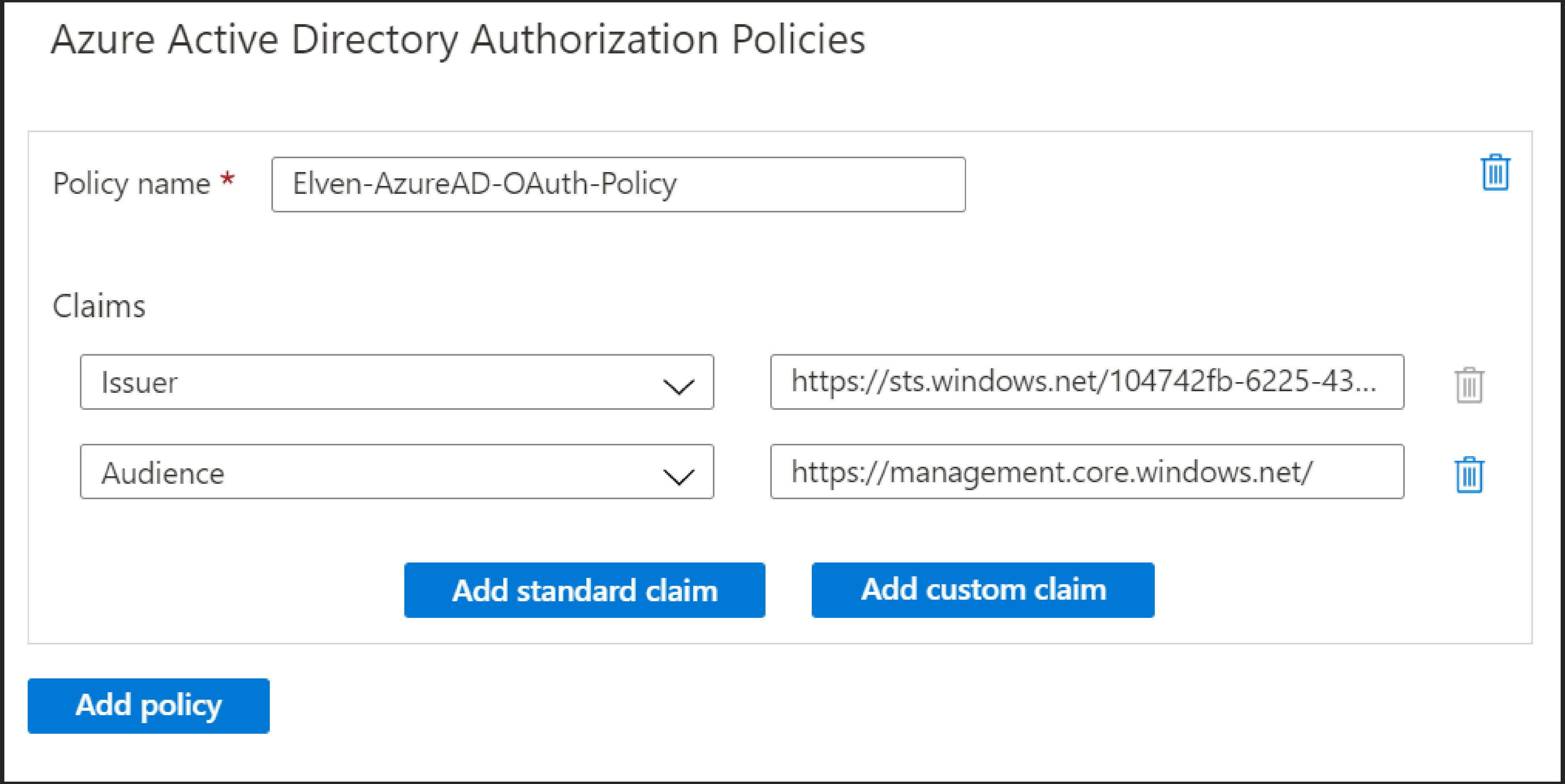
```
$accessToken = Get-AzAccessToken
$bearerToken = ConvertTo-SecureString ($accessToken.Token) -AsPlainText -Force
```



```
Invoke-RestMethod -Method Post -Uri $serverlessUrl -Authentication OAuth -Token $bearerToken
```

Logic App Authorization Policy

NORDIC
— VIRTUAL SUMMIT —



Azure Active Directory Authorization Policies

Policy name * Elven-AzureAD-OAuth-Policy

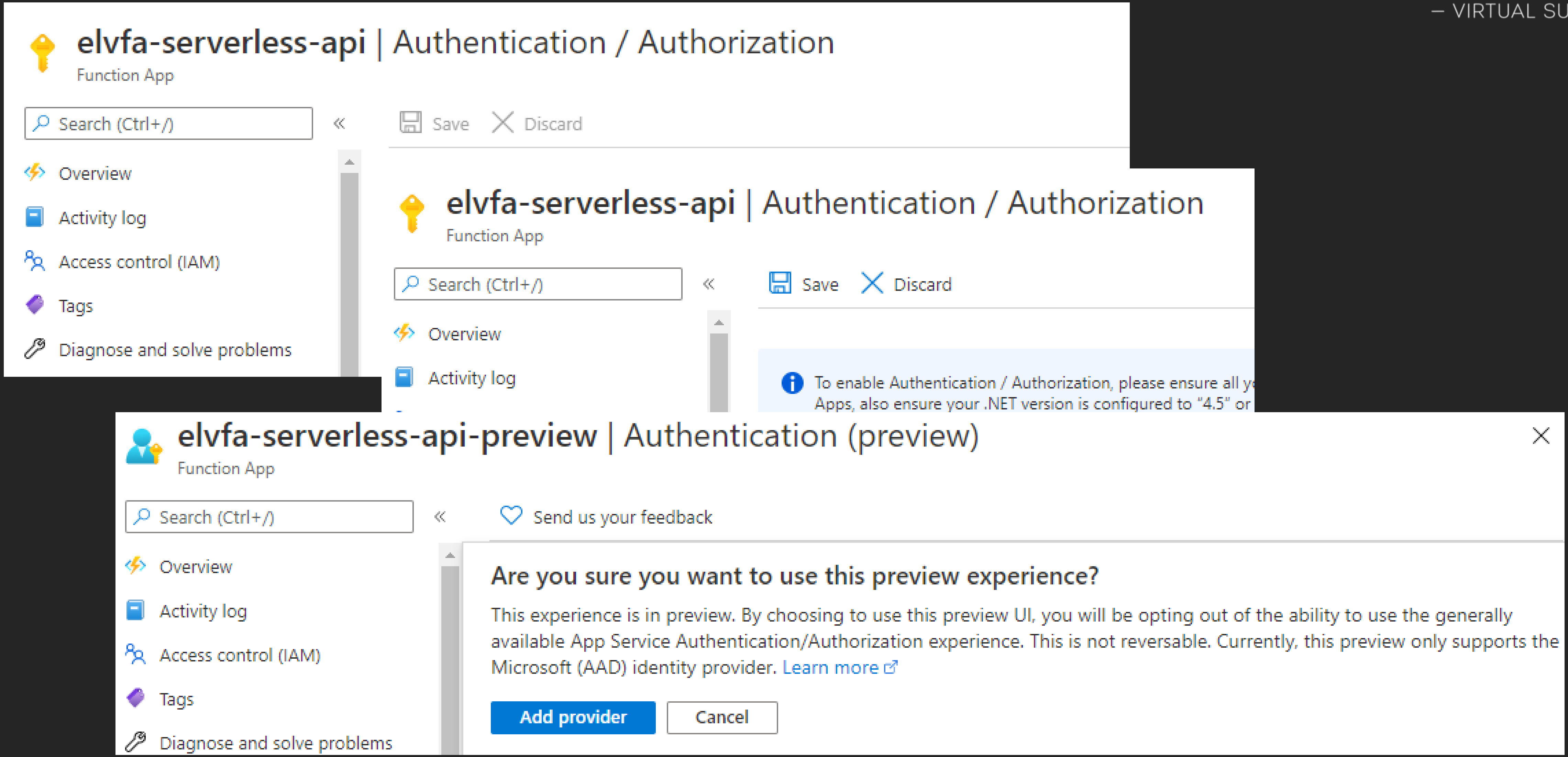
Claims

Issuer: https://sts.windows.net/104742fb-6225-43...

Audience: https://management.core.windows.net/

Azure Function Easy Auth

NORDIC — VIRTUAL SUMMIT —



elvfa-serverless-api | Authentication / Authorization

Function App

elvfa-serverless-api | Authentication / Authorization

Function App

To enable Authentication / Authorization, please ensure all y
Apps, also ensure your .NET version is configured to "4.5" or

elvfa-serverless-api-preview | Authentication (preview)

Function App

Are you sure you want to use this preview experience?

This experience is in preview. By choosing to use this preview UI, you will be opting out of the ability to use the generally available App Service Authentication/Authorization experience. This is not reversable. Currently, this preview only supports the Microsoft (AAD) identity provider. Learn more

Demo 1

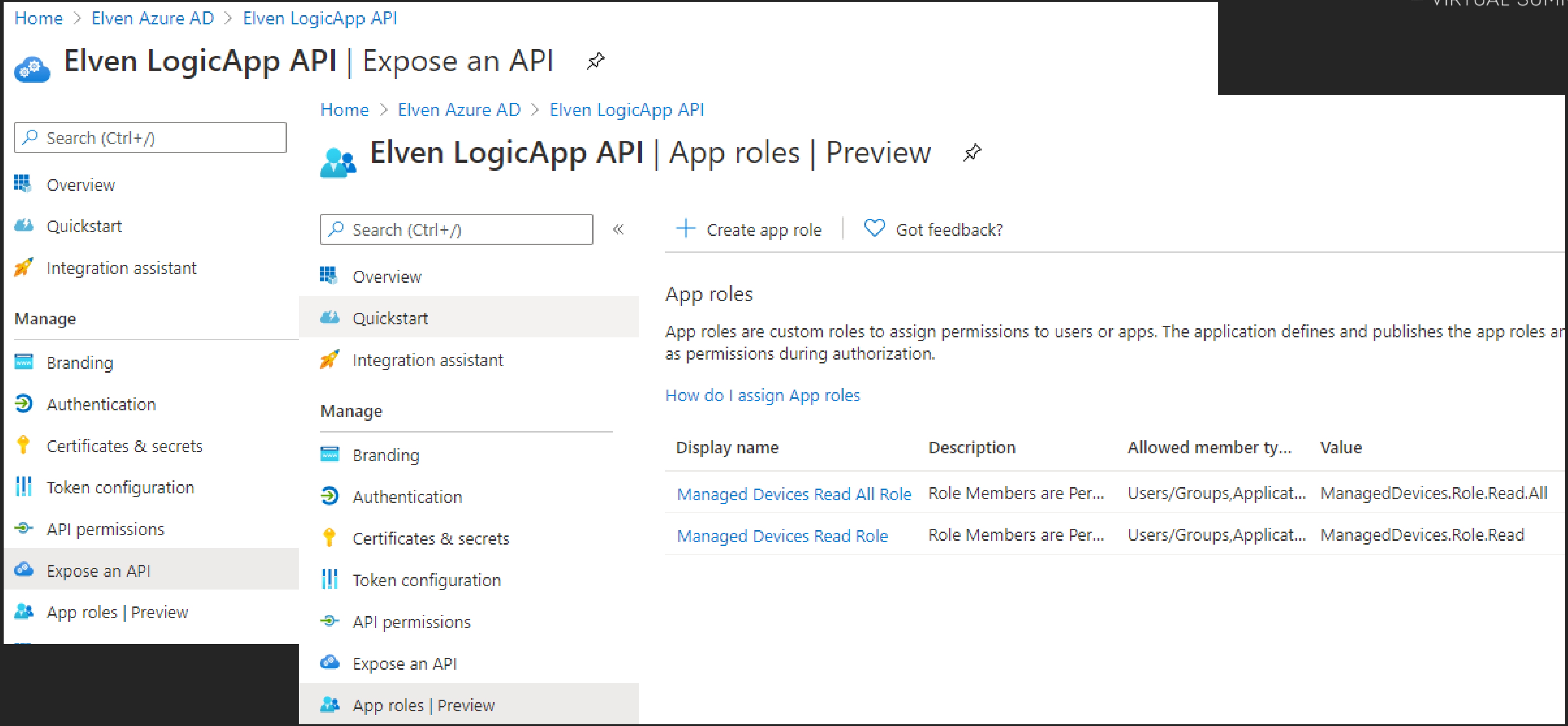
Scenario Management Access

Scenario – Custom API

Build and Expose your own Custom API using Azure AD

Using App Registrations for API

NORDIC
— VIRTUAL SUMMIT —



Elven LogicApp API | Expose an API

Elven LogicApp API | App roles | Preview

App roles

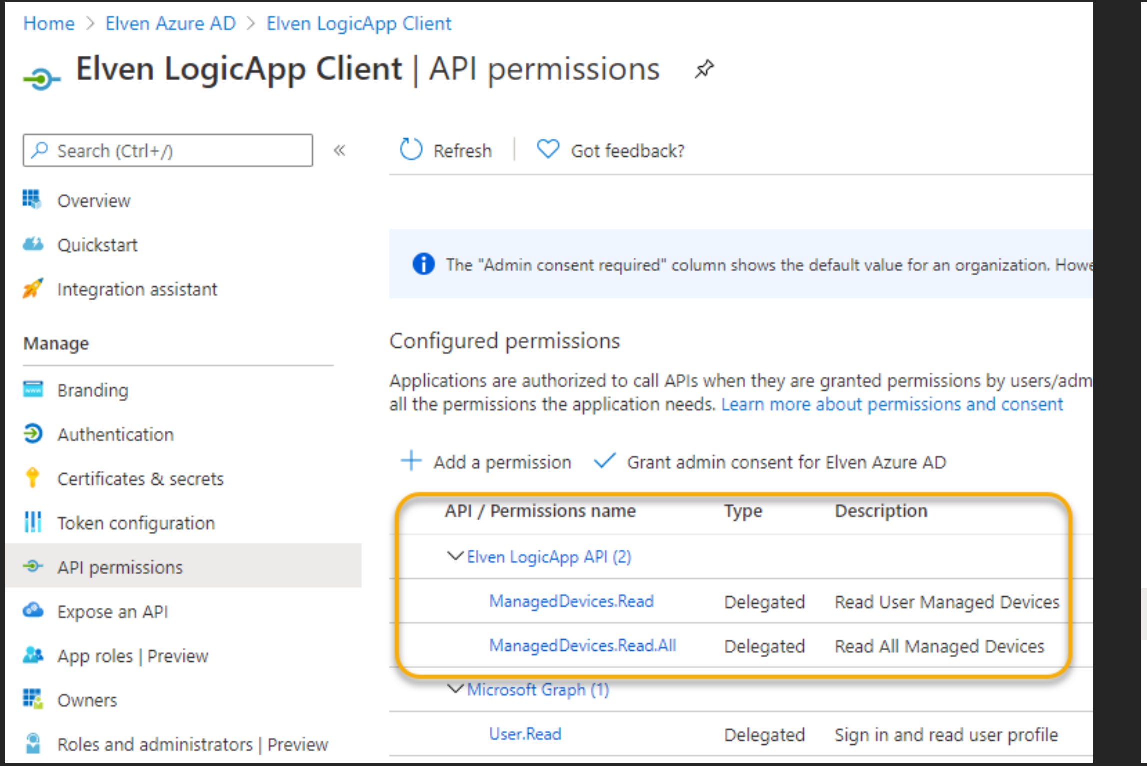
App roles are custom roles to assign permissions to users or apps. The application defines and publishes the app roles ar as permissions during authorization.

How do I assign App roles

Display name	Description	Allowed member ty...	Value
Managed Devices Read All Role	Role Members are Per...	Users/Groups,Applicat...	ManagedDevices.Role.Read.All
Managed Devices Read Role	Role Members are Per...	Users/Groups,Applicat...	ManagedDevices.Role.Read

App Registration for Client(s)

NORDIC
— VIRTUAL SUMMIT —



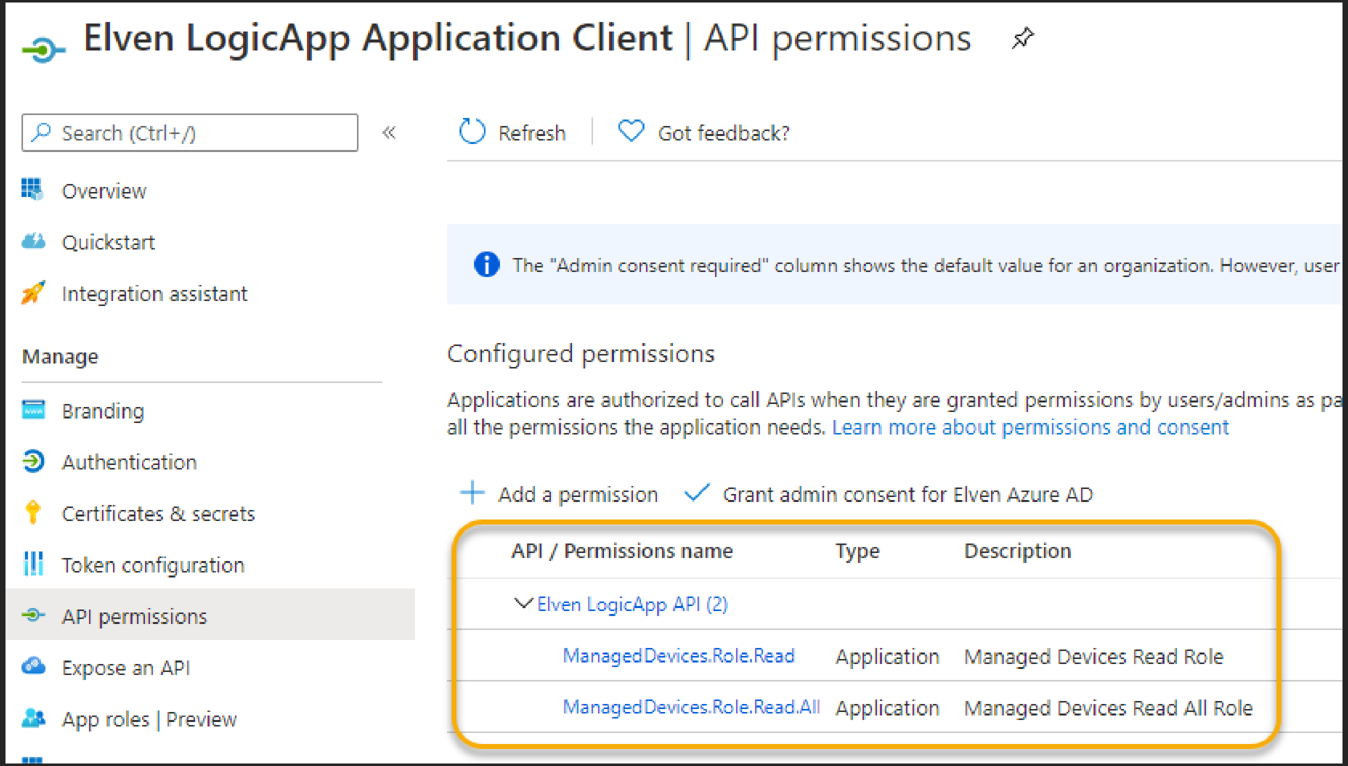
Elven LogicApp Client | API permissions

The "Admin consent required" column shows the default value for an organization. Howe

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/adm all the permissions the application needs. Learn more about permissions and consent

API / Permissions name	Type	Description
⌄Elven LogicApp API (2)		
ManagedDevices.Read	Delegated	Read User Managed Devices
ManagedDevices.Read.All	Delegated	Read All Managed Devices
⌄Microsoft Graph (1)		
User.Read	Delegated	Sign in and read user profile



Elven LogicApp Application Client | API permissions

The "Admin consent required" column shows the default value for an organization. However, user

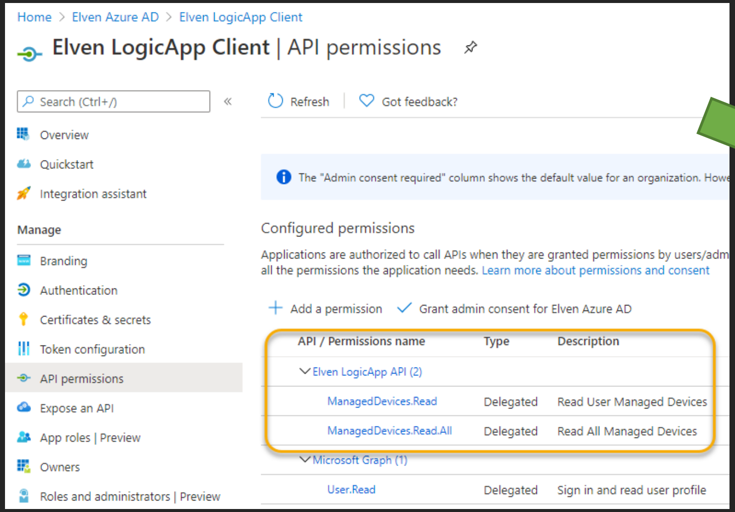
Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as pa all the permissions the application needs. Learn more about permissions and consent

API / Permissions name	Type	Description
⌄Elven LogicApp API (2)		
ManagedDevices.Role.Read	Application	Managed Devices Read Role
ManagedDevices.Role.Read.All	Application	Managed Devices Read All Role

App Registration API Strategy

NORDIC
— VIRTUAL SUMMIT —



Home > Elven Azure AD > Elven LogicApp Client

Elven LogicApp Client | API permissions

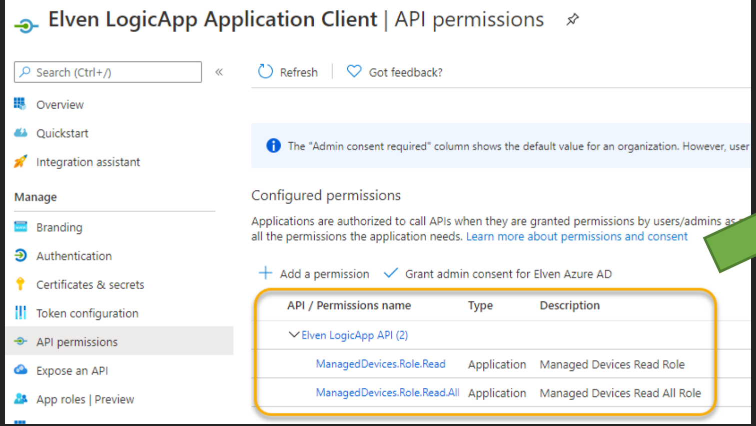
The "Admin consent required" column shows the default value for an organization. Howe

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/adm
all the permissions the application needs. Learn more about permissions and consent

+ Add a permission ✓ Grant admin consent for Elven Azure AD

API / Permissions name	Type	Description
Elven LogicApp API (2)		
ManagedDevices.Read	Delegated	Read User Managed Devices
ManagedDevices.Read.All	Delegated	Read All Managed Devices
Microsoft Graph (1)		
User.Read	Delegated	Sign in and read user profile



Elven LogicApp Application Client | API permissions

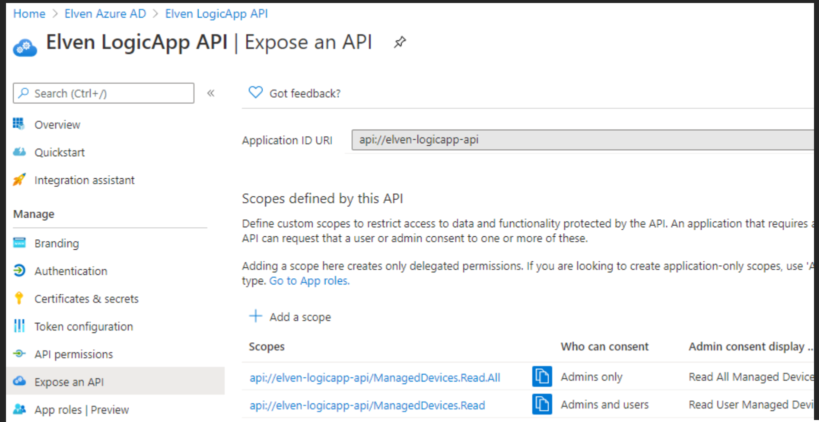
The "Admin consent required" column shows the default value for an organization. However, user

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as
all the permissions the application needs. Learn more about permissions and consent

+ Add a permission ✓ Grant admin consent for Elven Azure AD

API / Permissions name	Type	Description
Elven LogicApp API (2)		
ManagedDevices.Role.Read	Application	Managed Devices Read Role
ManagedDevices.Role.Read.All	Application	Managed Devices Read All Role



Home > Elven Azure AD > Elven LogicApp API

Elven LogicApp API | Expose an API

Application ID URI: api://elven-logicapp-api

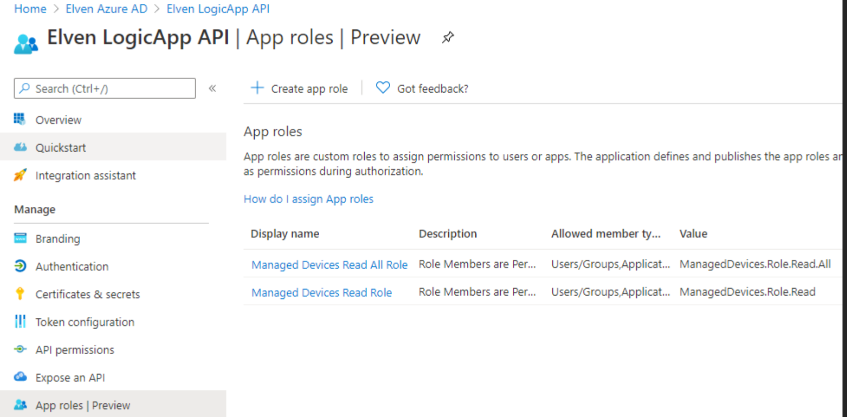
Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires a
API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create application-only scopes, use 'A
type. Go to App roles.

+ Add a scope

Scopes	Who can consent	Admin consent display ..
api://elven-logicapp-api/ManagedDevices.Read.All	Admins only	Read All Managed Device
api://elven-logicapp-api/ManagedDevices.Read	Admins and users	Read User Managed Devi



Home > Elven Azure AD > Elven LogicApp API

Elven LogicApp API | App roles | Preview

App roles

App roles are custom roles to assign permissions to users or apps. The application defines and publishes the app roles a
as permissions during authorization.

How do I assign App roles

Display name	Description	Allowed member ty...	Value
Managed Devices Read All Role	Role Members are Per...	Users/Groups,Applicat...	ManagedDevices.Role.Read.All
Managed Devices Read Role	Role Members are Per...	Users/Groups,Applicat...	ManagedDevices.Role.Read

Demo 2

API and Authenticating with App Registrations

Scenario Building Serverless API

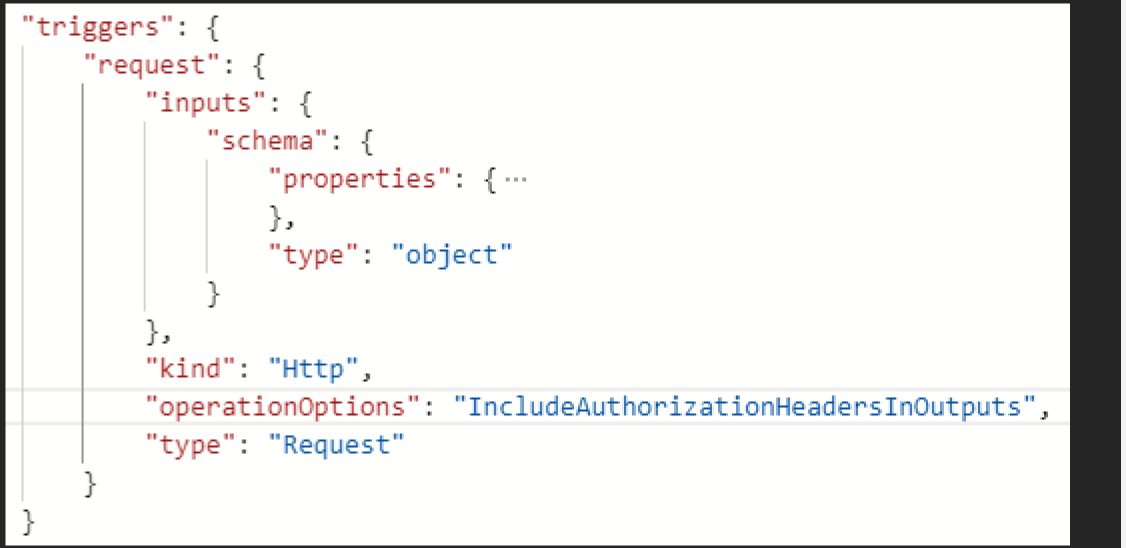
Building Serverless Solutions for Authentication and Authorization with Custom API

Serverless API Authentication & Authorization

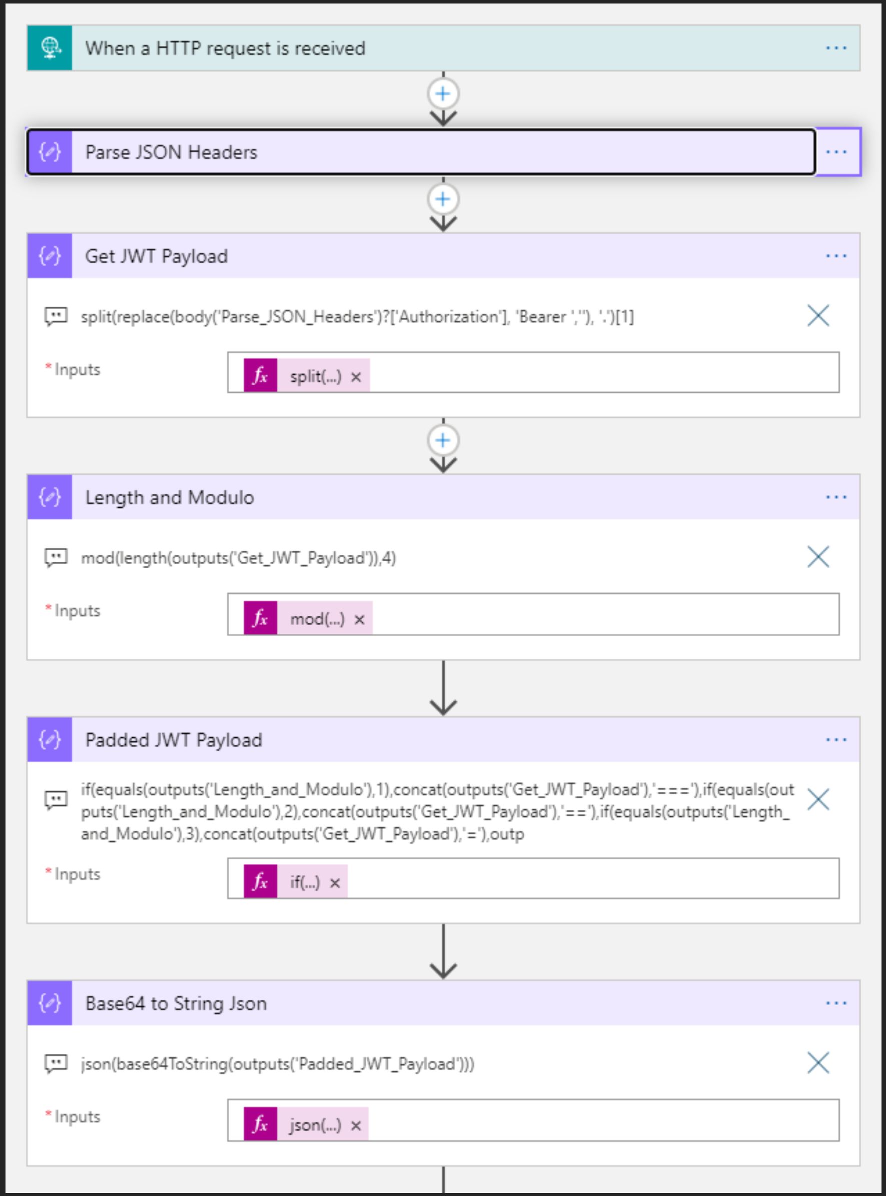
- 2 approaches:
    - Using one of or both

- Pre-Auth:
    - OAuth2 Authorization Policy/Easy Auth
    - Require matching Issuer and Audience
    - Logic App/Function App never triggers if not matching
- Inside Auth:
    - Checking of Claims inside App
    - Custom response handling

How to Include Authorization Header in Logic Apps



```
"triggers": {
    "request": {
        "inputs": {
            "schema": {
                "properties": {…
                },
                "type": "object"
            }
        },
        "kind": "Http",
        "operationOptions": "IncludeAuthorizationHeadersInOutputs",
        "type": "Request"
    }
}
```



When a HTTP request is received

Parse JSON Headers

Get JWT Payload

split(replace(body('Parse_JSON_Headers')?['Authorization'], 'Bearer ',''), '.')[1]

* Inputs: split(...)

Length and Modulo

mod(length(outputs('Get_JWT_Payload')),4)

* Inputs: mod(...)

Padded JWT Payload

if(equals(outputs('Length_and_Modulo'),1),concat(outputs('Get_JWT_Payload'),'==='),if(equals(outputs('Length_and_Modulo'),2),concat(outputs('Get_JWT_Payload'),'=='),if(equals(outputs('Length_and_Modulo'),3),concat(outputs('Get_JWT_Payload'),'='),outp

* Inputs: if(...)

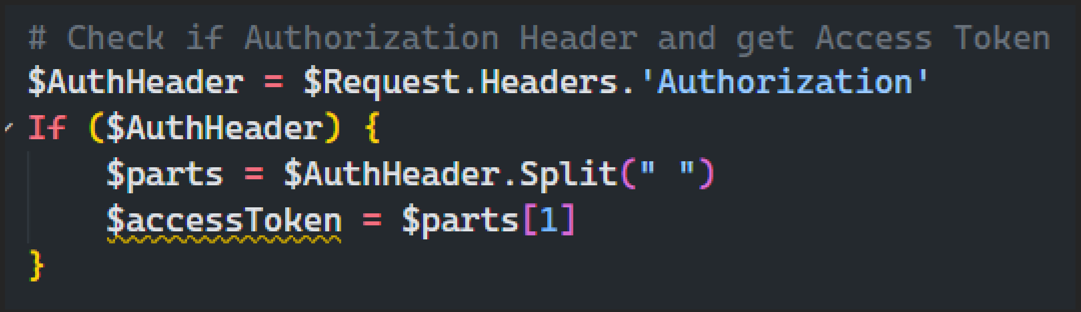
Base64 to String Json

json(base64ToString(outputs('Padded_JWT_Payload')))

* Inputs: json(...)

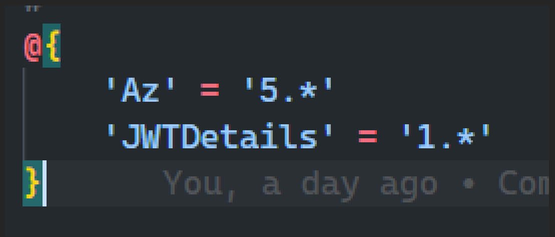
Get Authorization Header in Azure Functions

- Token will be in base64, so must be decoded



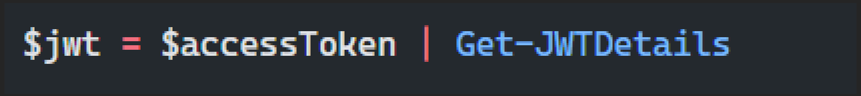
```
# Check if Authorization Header and get Access Token
$AuthHeader = $Request.Headers.'Authorization'
If ($AuthHeader) {
    $parts = $AuthHeader.Split(" ")
    $accessToken = $parts[1]
}
```

- In requirements.psd1:



```
@{
    'Az' = '5.*'
    'JWTDetails' = '1.*'
}       You, a day ago • Com
```

- In run.ps1:



```
$jwt = $accessToken | Get-JWTDetails
```

API calling another API

- Your serverless API can send requests to other APIs
  - Like Azure REST APIs, Microsoft Graph API etc
- Use Managed Identity for Application Permission (Roles)

Managed Service Identity (MSI)

- Can have Azure RBAC role assignments
  - Build Custom Roles where ever possible
  - Consider assign using PIM (active and time limited)

- Can have Microsoft Graph Application permissions
  - Assignment via Azure AD PowerShell only

Demo 3

Using Logic Apps and Azure Functions with OAuth2 Authorization

Summary & Resources

- Learning by doing
  - https://docs.microsoft.com/nb-no/learn/modules/getting-started-identity/
  - https://docs.microsoft.com/en-us/learn/paths/architect-api-integration/

- Demo resources
  - https://github.com/JanVidarElven/ProtectAzureServerLessWithAzureAD

- Blog posts
  - https://gotoguy.blog/2020/12/31/protect-logic-apps-with-azure-ad-oauth-part-1-management-access/
  - https://gotoguy.blog/2021/01/11/protect-logic-apps-with-azure-ad-oauth-part-2-expose-logic-app-as-api/
  - https://gotoguy.blog/2021/02/04/protect-logic-apps-with-azure-ad-oauth-part-3-connect-to-api-from-power-platform/

  - ++more in series coming (APIM, Azure Functions, etc)

- Connect and interact: @JanVidarElven ☺

Thank you!

NORDIC
– VIRTUAL SUMMIT –

MSEndPointMgr.com
#MSEndPointMgr

System Center User Group
Finland
#SCUGFI

System Center User Group
Denmark

#SCUGDK

System Center User Group
Sweden
#SCUGSE

Modern Management User Group
Norway
#MMUGNO

Please evaluate our session

NORDIC
— VIRTUAL SUMMIT —

https://2021.nordicvirtualsummit.com/feedback