/OUTSIDER SECURITY

BYO IDP in Entra ID

Persisting and bypassing MFA with OIDC based protocols

About me

/OUTSIDER SECURITY

- Dirk-jan Mollema
- Lives in The Netherlands
- Hacker / Researcher / Founder / Trainer @ Outsider Security
- Talks at Black Hat / Def Con / BlueHat / Troopers / x33fcon
- Author of several Active Directory and Entra ID tools
  - mitm6
  - Ldapdomaindump
  - adidnsdump
  - BloodHound.py
  - ntlmrelayx / krbrelayx
  - ROADtools

Socials

Blog/talks:	dirkjanm.io
Twitter/X:	@_dirkjan
BlueSky:	@dirkjanm.io

Talk agenda

- What is OIDC and why should we care
- OIDC and federated credentials
- Entra External Authentication Methods
- Conditional Access custom controls
- Detection opportunities and challenges

Entra ID

- Microsoft’s Identity-as-a-service platform
- Formerly called Azure AD
- Practically handles all cloud-based sign-ins in modern enterprises

Why OpenID connect

- Most of the time we want to have access tokens / bearer tokens
- They give us access to data
- Focus often on Microsoft 365 native apps
- Little interest in how the actual validation of these tokens works

Why OpenID connect

- Whereas access tokens provide **authorization** for resources, ID tokens provide **authentication** to clients
- Web and native apps can use ID tokens to authenticate a user
- Used extensively for SSO with Entra ID

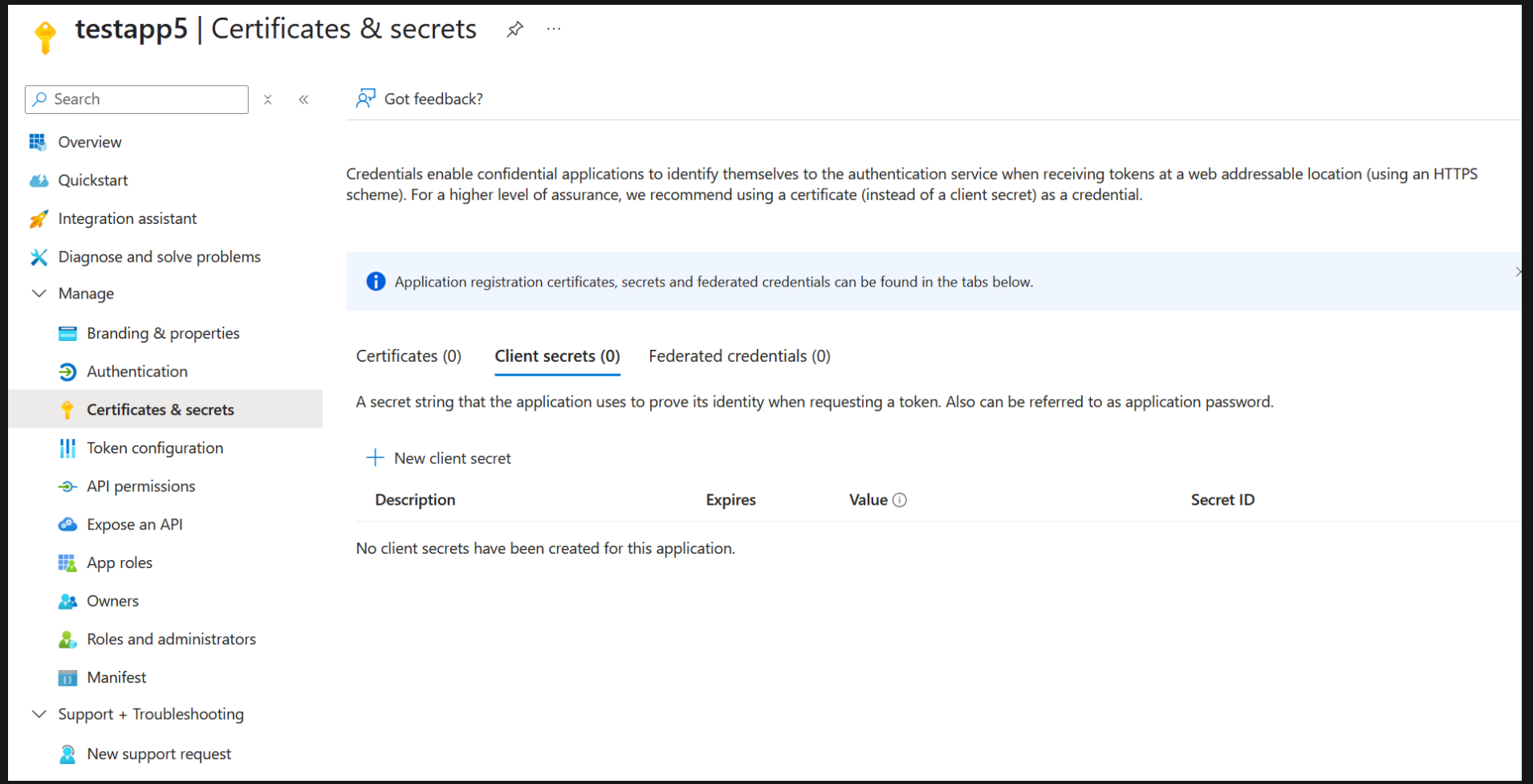
OIDC technical bits

- Based on OAuth2
- The Authorization Server (often Entra ID) will issue an ID token during the familiar OAuth flows
- ID token requested by openid scope

- Main difference with access token:
  - Access token is intended for the Resource Provider (upstream API such as Microsoft Graph) to authorize the user
  - ID token is intended for the client to authenticate the user

Federated credentials

Application credentials



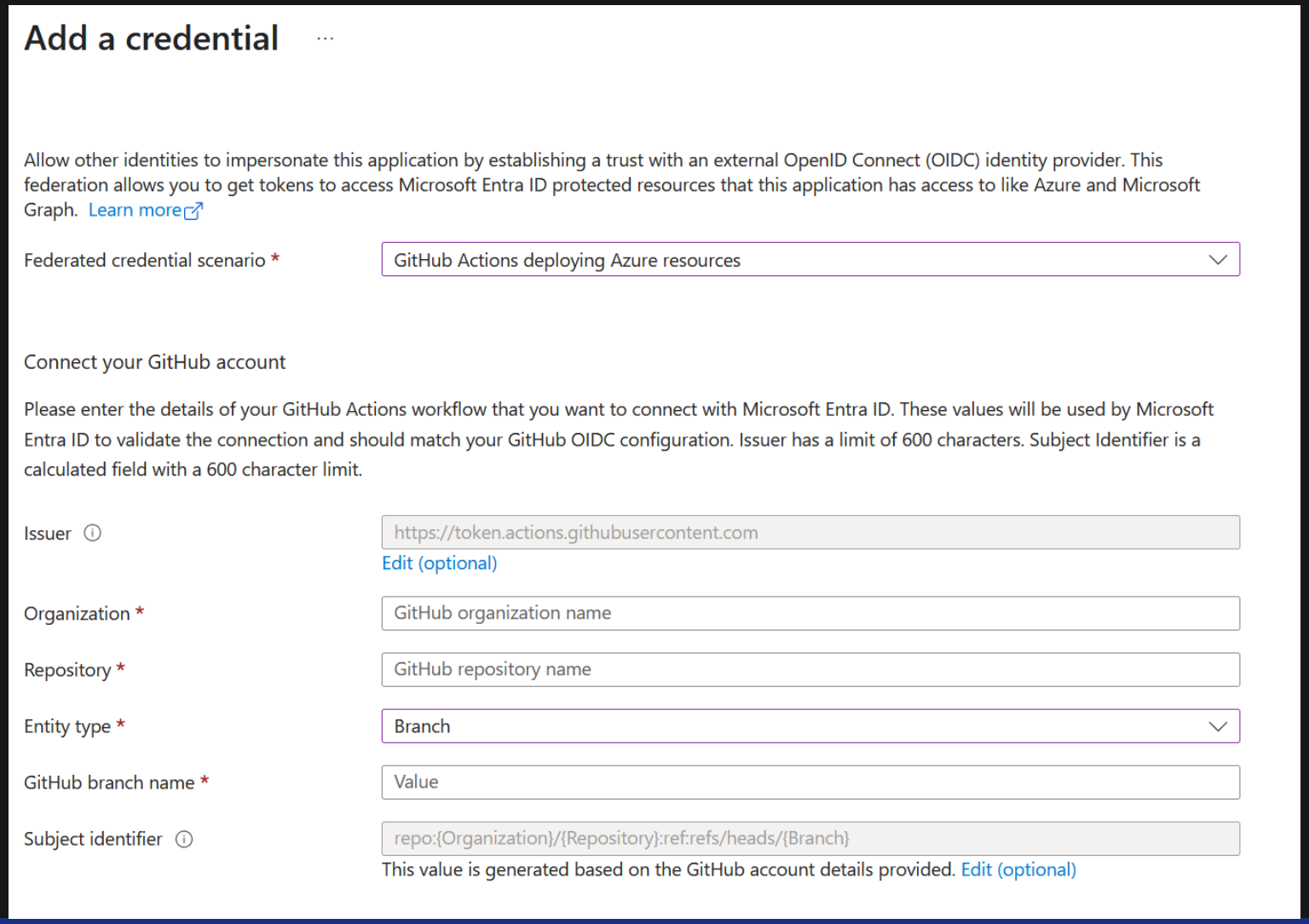
Application credentials

- Client secrets or passwords
  - Use the client secret itself to auth
- Certificates + private key
  - Use signed assertion to authenticate
- Federated Credentials
  - Allow another IDP to authenticate?

Federated credentials

- Federation = establishing a trust relationship between an application (Service Provider or Resource Provider) and an Identity Provider so that the IdP can authenticate users to your app.
- Federation = making something else responsible for your authentication and trusting they do their job properly.
- Best known example: AD FS
- Essentially trusting another IdP (not Entra ID) to issue tokens so we can get tokens from the IdP we want (Entra ID)

Configuring federated credentials



Custom providers

Add a credential

Allow other identities to impersonate this application by establishing a trust with an external OpenID Connect (OIDC) identity provider. This federation allows you to get tokens to access Microsoft Entra ID protected resources that this application has access to like Azure and Microsoft Graph. Learn more

Federated credential scenario * Other issuer

Connect your account

Enter the details of the account that you want to connect with Microsoft Entra ID. These values will be used by Microsoft Entra ID to validate the connection.

Issuer * Issuer URL (Limit of 600 characters)

Type Explicit subject identifier Claims matching expression (Preview)

Value * Subject (Limit of 600 characters)

The subject identifier field is required

Federated credentials auth

- Uses OAuth2 client credentials flow with signed assertion

Third case: Access token request with a federated credential

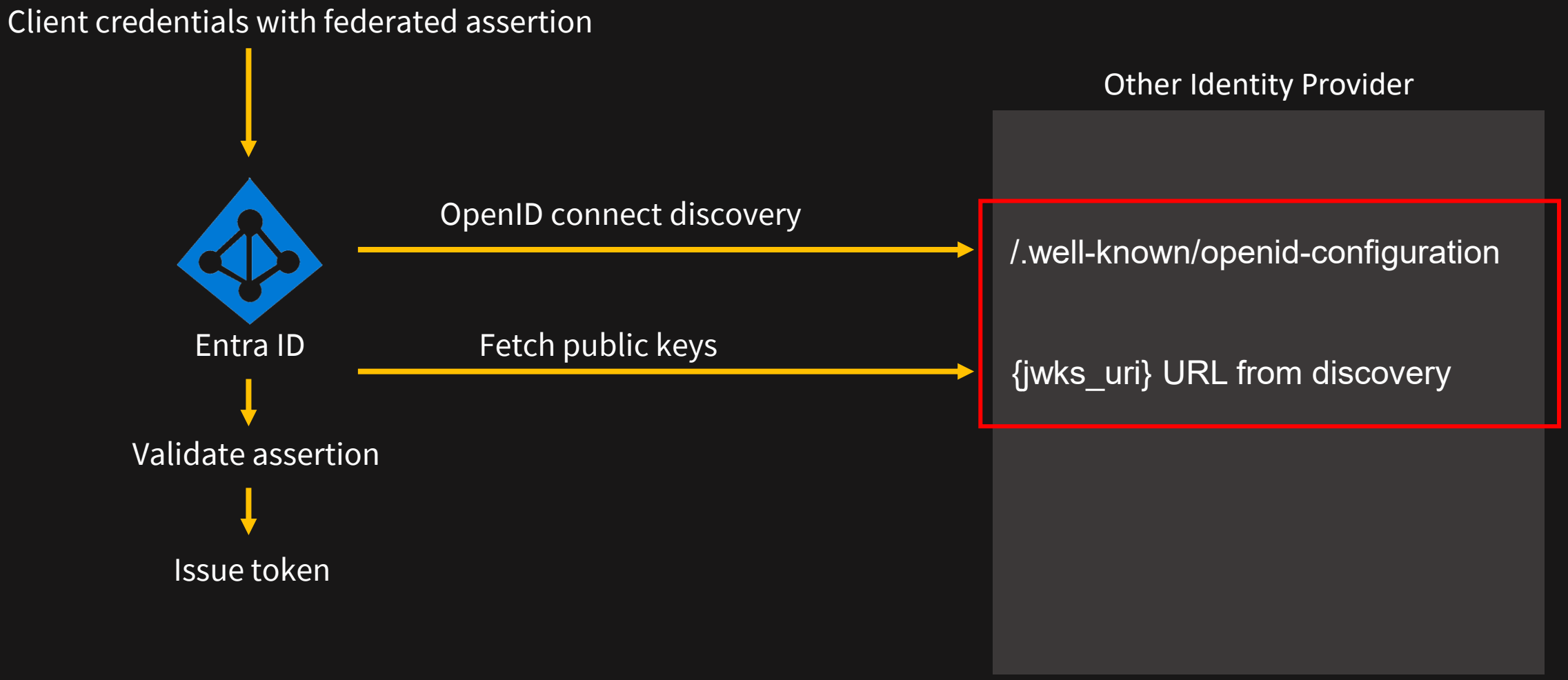
HTTP Copy

```
POST /{tenant}/oauth2/v2.0/token HTTP/1.1              // Line breaks for clarity
Host: login.microsoftonline.com:443
Content-Type: application/x-www-form-urlencoded

scope=https%3A%2F%2Fgraph.microsoft.com%2F.default
&client_id=11112222-bbbb-3333-cccc-4444dddd5555
&client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer
&client_assertion=eyJhbGciOiJSUzI1NiIsIng1dCI6Imd4OHRHeXN5amNScUtqRlBuZDdSRnd2d1pJMCJ9.eyJ{a lot of characte
&grant_type=client_credentials
```

Expand table

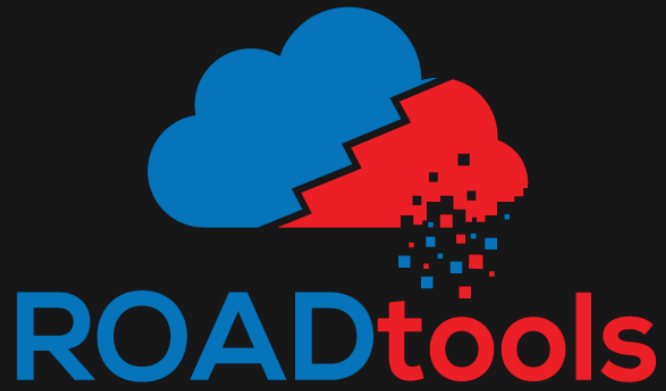
Parameter	Condition	Description
`client_assertion`	Required	An assertion (a JWT, or JSON web token) that your application gets from another identity provider outside of Microsoft identity platform, like Kubernetes. The specifics of this JWT must be registered on your application as a federated identity credential. Read about workload identity federation to learn how to setup and use assertions generated from other identity providers.

Federated credentials protocol

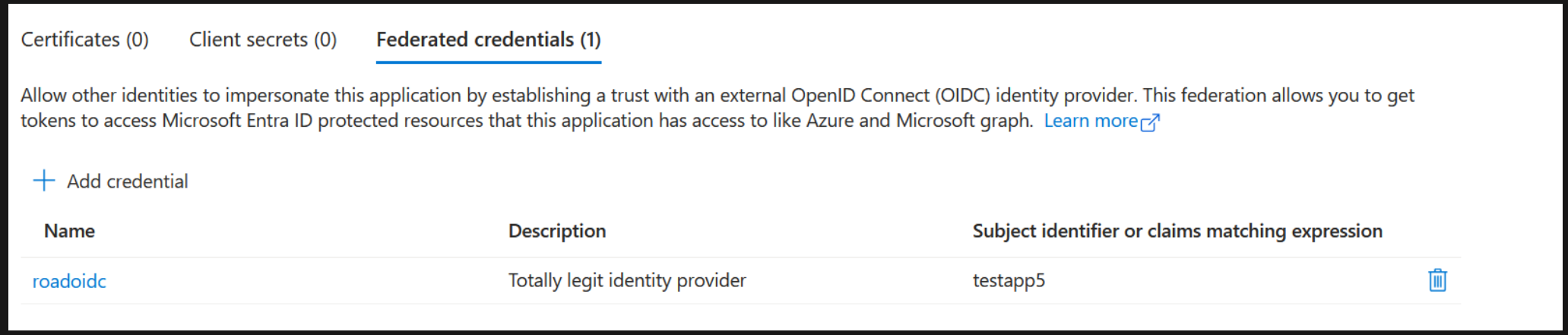


roadoidc

- Minimalistic OIDC implementation
- Can be hosted on Azure App services or Azure Blob storage
  - Azure Blob storage hosts static files, suitable for federated credentials but not for advanced scenarios we cover later
- Tokens can be requested with roadtx



Authenticating with federated creds




```
(ROADtools) → roadoidc git:(master) ✗ roadtx federatedappauth -c 2dfa5a93-3400-4b81-b6d4-2123c6c1ae6e --cert-pem roadoidc.pem --key-pem roadoidc.key
--subject testapp5 -t iminyour.cloud --issuer https://roadoidcblob.blob.core.windows.net/containername -s https://graph.microsoft.com/.default
Requesting token with scope https://graph.microsoft.com/.default
Tokens were written to .roadtools_auth
```

What's the point

- Mainly OPSEC
  - Adding client secrets and certificates to apps is a well-known technique and included in many detection playbooks.
  - Federated credentials are less well known by defenders and may not be spotted.

Adding client secret

TimeGenerated [UTC]	ResourceId	OperationName	OperationVersion	Category
6/11/2025, 8:03:07.864 AM	/tenants/6287f28f-4f7f-4322-9...	Update application	1.0	ApplicationManagement
6/11/2025, 8:03:07.863 AM	/tenants/6287f28f-4f7f-4322-9...	Update application – Certificates and secrets management	1.0	ApplicationManagement
6/11/2025, 8:03:07.799 AM	/tenants/6287f28f-4f7f-4322-9...	Update service principal	1.0	ApplicationManagement

Adding federated credentials

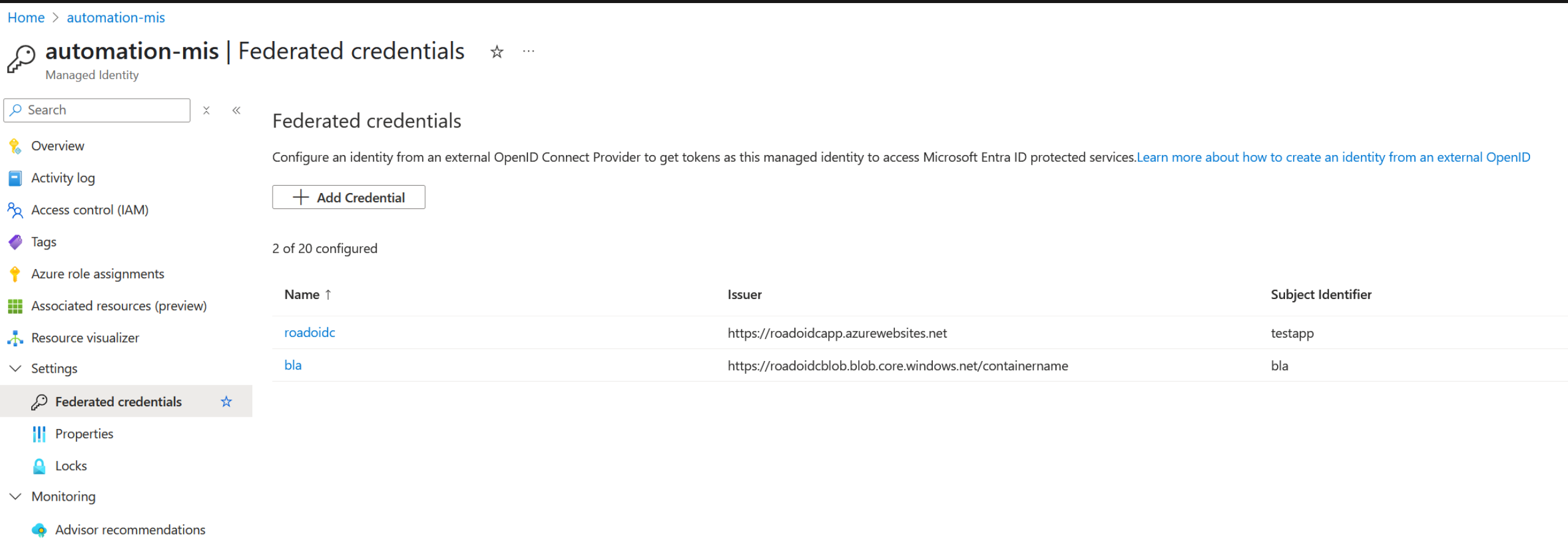
6/11/2025, 7:57:51.389 AM	/tenants/6287f28f-4f7f-4322-9...	Update application	1.0	ApplicationManagement
6/11/2025, 7:57:51.328 AM	/tenants/6287f28f-4f7f-4322-9...	Update service principal	1.0	ApplicationManagement

Federated credentials in Azure

- Federated credentials exist on User Managed Identities
- Normally, managed identities can only be accessed via resources they are linked to
  - Via Metadata endpoint on Virtual Machines, Logic Apps, etc
  - Gives out only access tokens, no long-term persistence without resource access
- Federated credentials allow for persistent access
  - Permanent credentials that can be used at any time
  - Can be used outside of Azure

More details: https://dirkjanm.io/persisting-with-federated-credentials-entra-apps-managed-identities/

Federated credentials on MI



OIDC in External Auth Methods

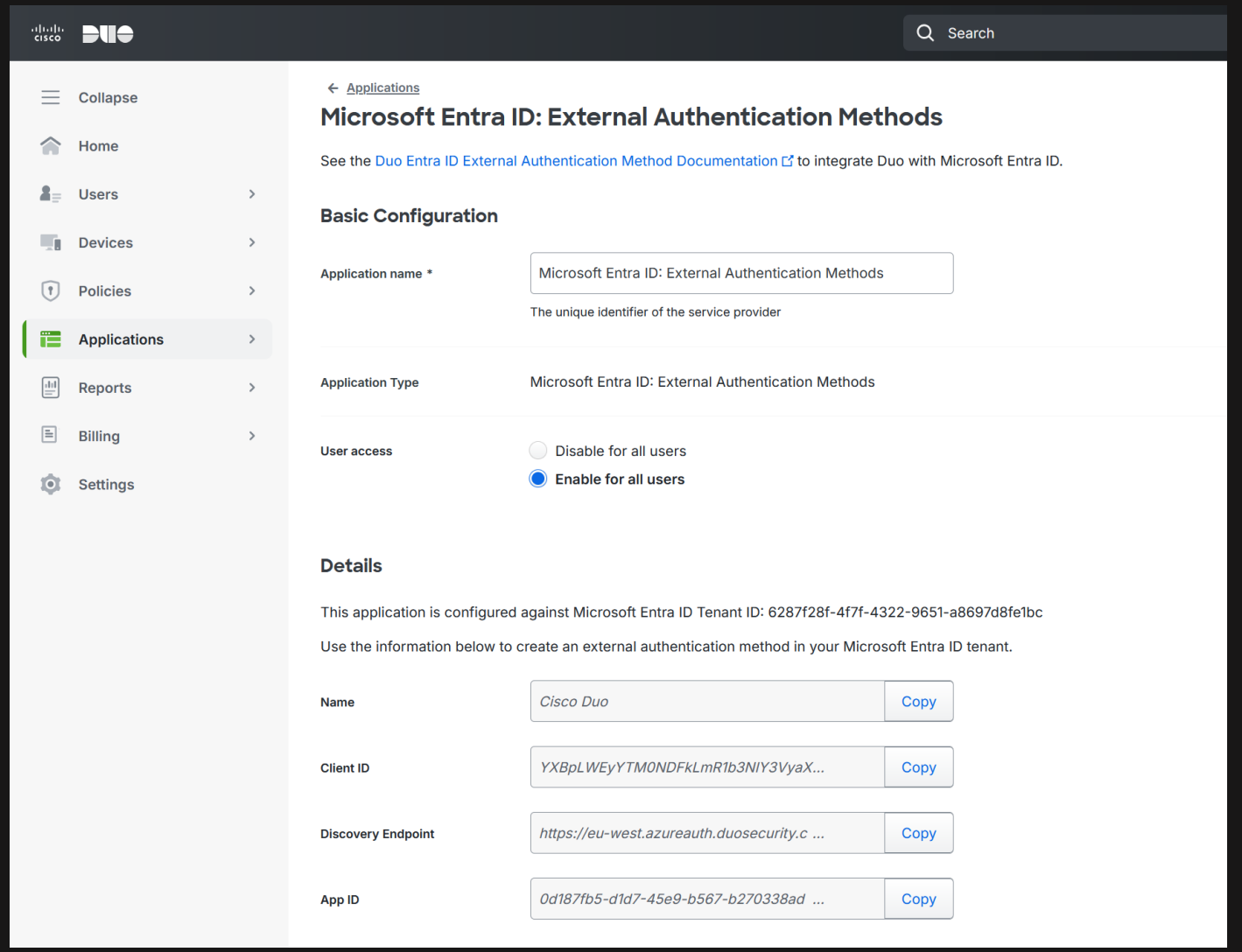
External Authentication Methods

- New-ish feature (May 2024) that makes it possible to use external MFA providers.
- Alternative for Entra ID native MFA.
- Uses OpenID Connect to trust authentication claims from IdPs

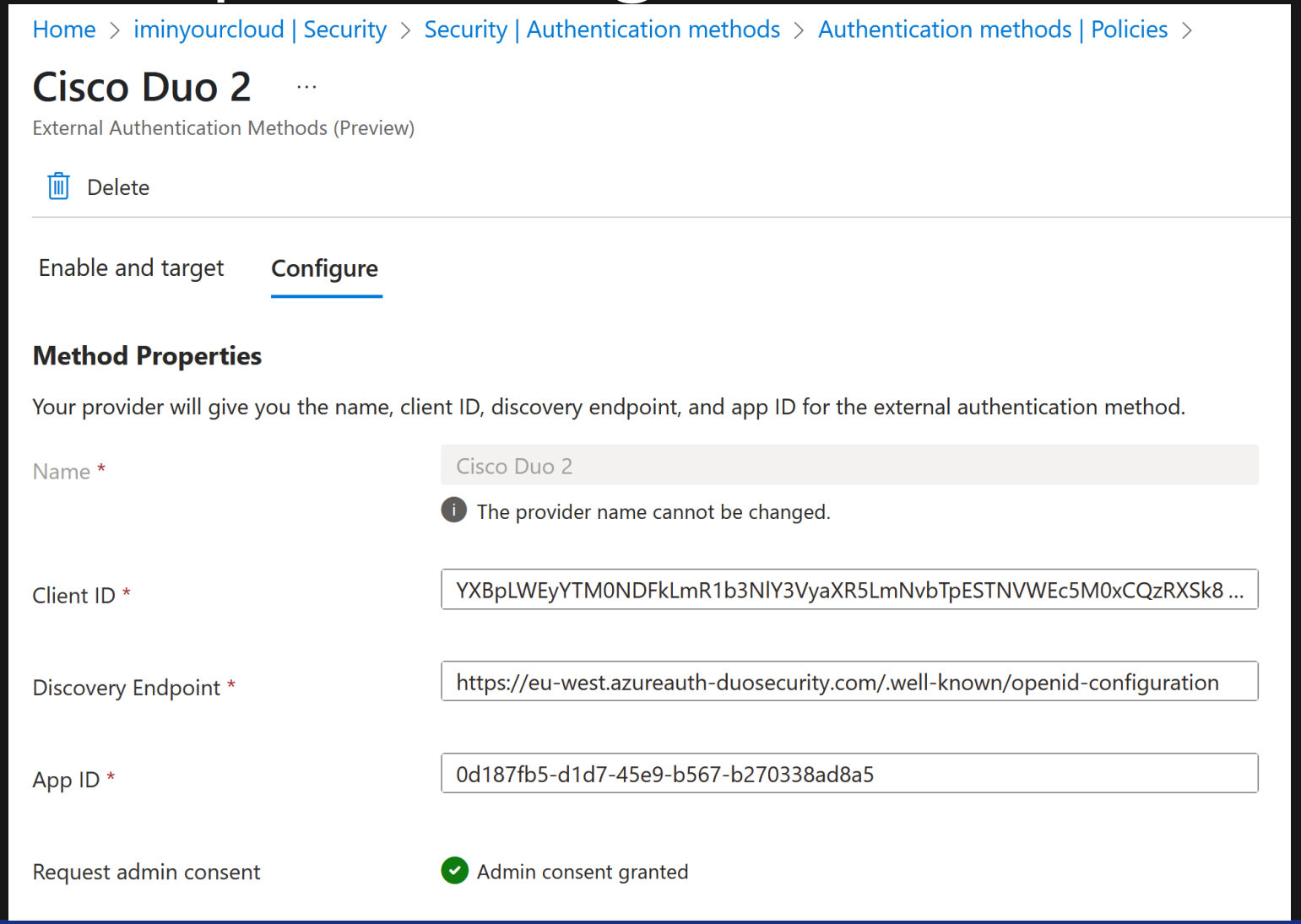
Reference: https://techcommunity.microsoft.com/blog/microsoft-entra-blog/public-preview-external-authentication-methods-in-microsoft-entra-id/4078808

Test setup

- Shout-out to DUO for free tier!

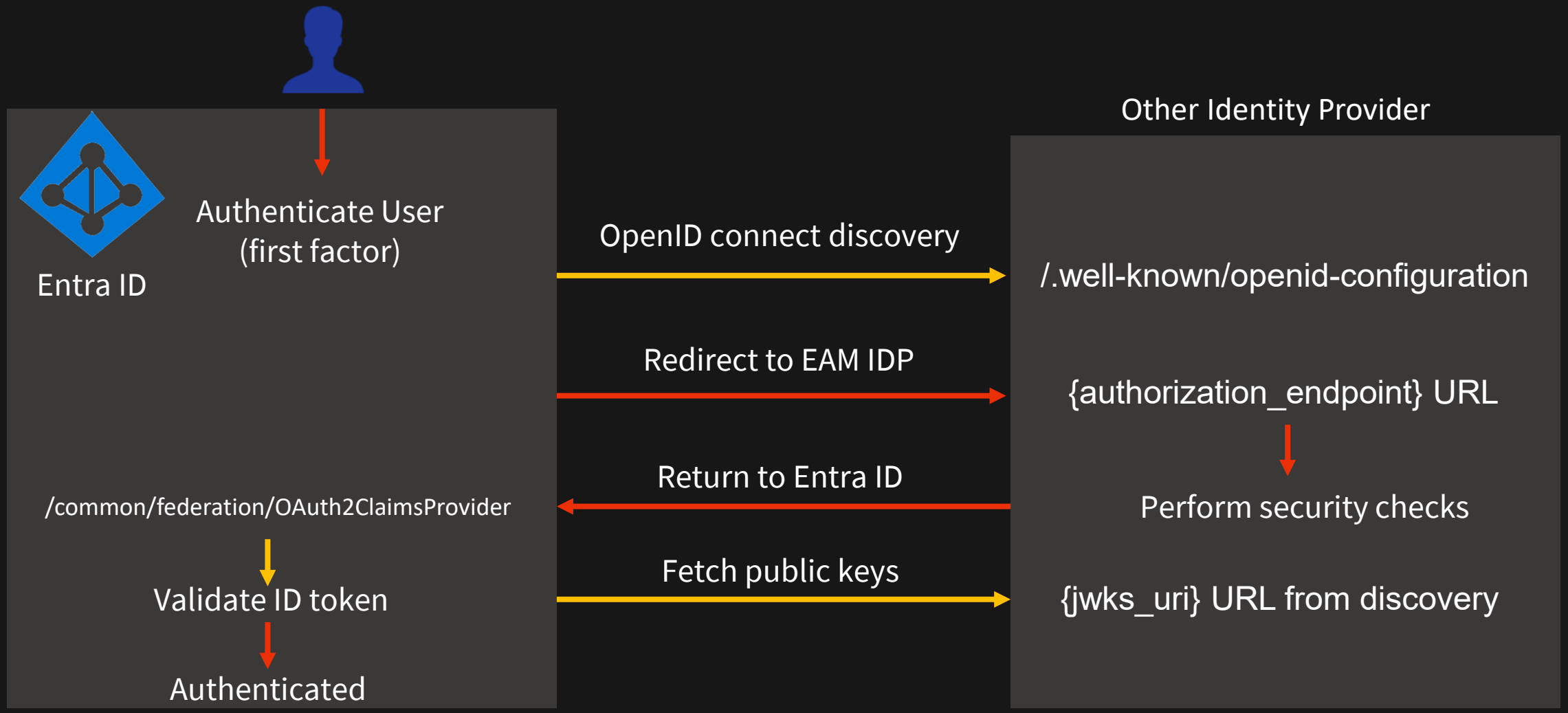


Test setup – configure EAM



EAM and OIDC

Gold = Entra backend request
Red = User flow



EAM request

Request

Pretty Raw Hex

```
POST /authorization HTTP/1.1
Host: eu-west.azureauth.duosecurity.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/20100101 Firefox/136.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://login.microsoftonline.com/
Content-Type: application/x-www-form-urlencoded
Content-Length: 2424
Origin: https://login.microsoftonline.com
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site
Priority: u=0, i
Te: trailers
Connection: keep-alive

scope=openid&response_mode=form_post&response_type=id_token&client_id=
YXBpLWEyYTM0NDFkLmR1b3NlY3VyaXR5LmNvbTpESTNVWEc5M0xCQzRXSk80OExOUQ%3D%3D&redirect_uri=
https%3A%2F%2Flogin.microsoftonline.com%2Fcommon%2Ffederation%2Fexternalauthprovider&claims=
%7B%22id_token%22%3A%7B%22amr%22%3A%7B%22essential%22%3Atrue%2C%22values%22%3A%5B%22face%22%2C%22fido%22%2C
%22fpt%22%2C%22hwk%22%2C%22iris%22%2C%22otp%22%2C%22tel%22%2C%22pop%22%2C%22retina%22%2C%22sc%22%2C%22sms%2
2%2C%22swk%22%2C%22vbm%22%2C%22bio%22%5D%7D%2C%22acr%22%3A%7B%22essential%22%3Atrue%2C%22values%22%3A%5B%22
possessionorinherence%22%5D%7D%7D%7D&nonce=
AwABEgEAAAADAOz_BQD0_7pmdPujlwS0nbEHB2JRGEuibnS2erMtZByN-uBXaemX7_7taa1LTL_pBec_Hr7SG5Y0A6oobvoOmO7i2s0KRrk
gAA&id_token_hint=
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IkNOdjBPSTNSd3FsSEZFVm5hb01Bc2hDSDJYRSJ9.eyJhdWQiOiIwZDE4N2ZiNS
1kMWQ3LTQ1ZTktYjU2Ny1iMjcwMzM4YWQ4YTUiLCJpc3MiOiJodHRwczovL2xvZ2luLm1pY3Jvc29mdG9ubGluZS5jb20vNjI4N2YyOGYtN
GY3Zi00MzIyLTk2NTEtYTg2OTdkOGZlMWJjL3YyLjAiLCJpYXQiOjE3NDk3MjcyMTIsIm5iZiI6MTc0OTcyNzIxMiwiZXhwIjoxNzQ5NzI4
MTEyLCJuYW1lIjoiZHVvMyIsIm9pZCI6IjE4Y2JkZjFiLWU2OTQtNGU1MC1hZTU2LTkxNTYyMzU1NzM5ZiIsInByZWZlcnJlZF91c2VybmF
tZSI6ImR1bzNAaW1pbnlvdXIuY2xvdWQiLCJzdWIiOiJvZ0pHcUZoUENGZjMwb3gxUnZWTmVfUk9pV1dfazcyNTFZSlBlYVRvak80IiwidG
lkIjoiNjI4N2YyOGYtNGY3Zi00MzIyLTk2NTEtYTg2OTdkOGZlMWJjIiwidXBuIjoiZHVvM0BpbWlueW91ci5jbG91ZCIsInV0aSI6IlRxY
nVXRXZUMVVLZmJjNTdkQzhHQUEiLCJ2ZXIiOiIyLjAifQ.A25QsnXRxHtbcdXo_HMx34eKL6eQobfM8n8kf4K_W3fu7B6ysJ8G4bcvhX9Eg
IlBX5cNgaZ7BPwm5oV7dTdmKIPvHsNfZ1DxVt373_SG7_ccQTFg824z9mm_cndRPA7vtM9aruO-rJlyw2iKxrCZTzXxNQjqHtmlFiePhSyO
RDUcjs0pq3L-Gt5qxvNMcjgliW-8KH_-MVGlVkXc_VaclosyOpUuOv3xEB3R7UAI65ldR-YcT2T9FlEgTh73rKSftsTbrhMRSQP8I4B1CBi
Q9VmW2i3ty05VRWuLvy2EwjLUkhDCMmT2SDRoQ9rAY5IYkgV0KbDpoljYw6m4hBsEug&client-request-id=
728685a8-3bd8-4235-a494-68139b818e55&state=
```

ID token signed by Entra ID

HEADER: ALGORITHM & TOKEN TYPE


```
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "CNv0OI3RwqlHFEVnaoMAshCH2XE"
}
```


PAYLOAD: DATA


```
{
  "aud": "0d187fb5-d1d7-45e9-b567-b270338ad8a5",
  "iss": "https://
login.microsoftonline.com/6287f28f-4f7f-4322-9651-
a8697d8fe1bc/v2.0",
  "iat": 1749727212,
  "nbf": 1749727212,
  "exp": 1749728112,
  "name": "duo3",
  "oid": "18cbdf1b-e694-4e50-ae56-91562355739f",
  "preferred_username": "duo3@iminyour.cloud",
  "sub": "ogJGqFhPCFf30ox1RvVNe_ROiWW_k7251YJPeaToj04",
  "tid": "6287f28f-4f7f-4322-9651-a8697d8fe1bc",
  "upn": "duo3@iminyour.cloud",
  "uti": "TqbuWEvT1UKfbc57dC8GAA",
  "ver": "2.0"
}
```

EAM return result

Request

Pretty Raw Hex

```
POST /common/federation/externalauthprovider HTTP/1.1
Host: login.microsoftonline.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/20100101 Firefox/136.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 1979
Origin: https://eu-west.azureauth.duosecurity.com
Referer: https://eu-west.azureauth.duosecurity.com/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site
Priority: u=0, i
Te: trailers
Connection: keep-alive

token_type=Bearer&id_token=
eyJhbGciOiJSUzI1NiIsImtpZCI6IjZvRVFhZy1penIzRFV1WjNGd3JjcTFTMVhiay1ta054UkdYMm1JcDVraU0iLCJ0eXAiOiJKV1QifQ.
eyJpc3MiOiJodHRwczovL2V1LXdlc3QuYXp1cmVhdXRoLmR1b3NlY3VyaXR5LmNvbSIsInN1YiI6Im9nSkdxRmhQQ0ZmMzBveDFSdlZOZV9
ST2lXV19rNzI1MVlKUGVhVG9qTzQiLCJhdWQiOiJZWEJwTFdFeVlUTTBOREZrTG1SMWIzTmxZM1Z5YVhSNUxtTnZiVHBFU1ROVldFYzVNMH
hDUXpSWFNrODBPRXhPVVE9PSIsImV4cCI6MTc0OTcyNzg1OSwiaWF0IjoxNzQ5NzI3NTU5LCJub25jZSI6IkF3QUJFZ0VBQUFBREFPel9CU
UQwXzdubWRQdWpKd1MwbmhGSEIySlJHRnVpYm5TMmVyTXRaQnlOLXVCWGFlbVg3Xzd0YWExTElMX3BCZWNfSHI3U0c1WTBBNm9vYnZvT21P
N2kyczBLUnJrZ0FBIiwiRHVvTWZhIjoiTWZhRG9uZSIsImFjciI6InBvc3Nlc3Npb25vcmluaGVyZW5jZSIsImFtciI6WyJwb3AiXX0.dgQ
Rx6NhnRFfORFQqVxzRx0kksTrjrRnHESPFbGrQIuSS-aRGhsUl_v0fNA-Vg0CVIbtkijQT_uL5r0APXubNDARYlhW0xd7taIiqZxVgFV-97
FpJc-glOls38dKSCUnIsNJ4zI5oUal_352n3BjsDe0fHxSn8FnGN_gh-ugcXkMdq31E1BmG_CG8Ou1yRInnbNnpol3DZUcsA9ktZ2EWemJs
gZsVjklZbxUH9xs1SZdNWFWFVBxy7BKopGuT3wYJ2c002GMS8j6QlTIKotqZg1TgMCRySbuTlgatR-LWFII9-HiFcm7ZINiP9yzYpz9WQob
2h-lX2lXwODvvbbQBkCH6X8rAdLCXIwEtcVEYfdH3Qyh_RcvpK-ncSA4ByYrFO8oTAy5eRxHHh9mgKS1Tg8ZqPLV3gNImNyQ22v_aZQfqmT
2pAnuFQiSv3bjVeNRdgVSwDiKHsPcpyFCQ__fEmSyrEW-txWicdxbiFgurU2ybM3qEqtMrjnYPFf6v4bcyFWsy-P9TX1o1tbvQ580vYMr6S
NfONwtgLUu7aOwheN4ByNNH-Zla9--Tfe2swGDr8t0-nGo7GIspuBeAXarfVJMu53pQZm584TlhTU4cC0dOBbikIC5uuOma-Epmh531Cq91
gfCj4tqaopN41mYL3ZNejwbgUH-lJ3dhw285B8&expires_in=300&state=
```

ID token signed by EAM IDP

HEADER: ALGORITHM & TOKEN TYPE

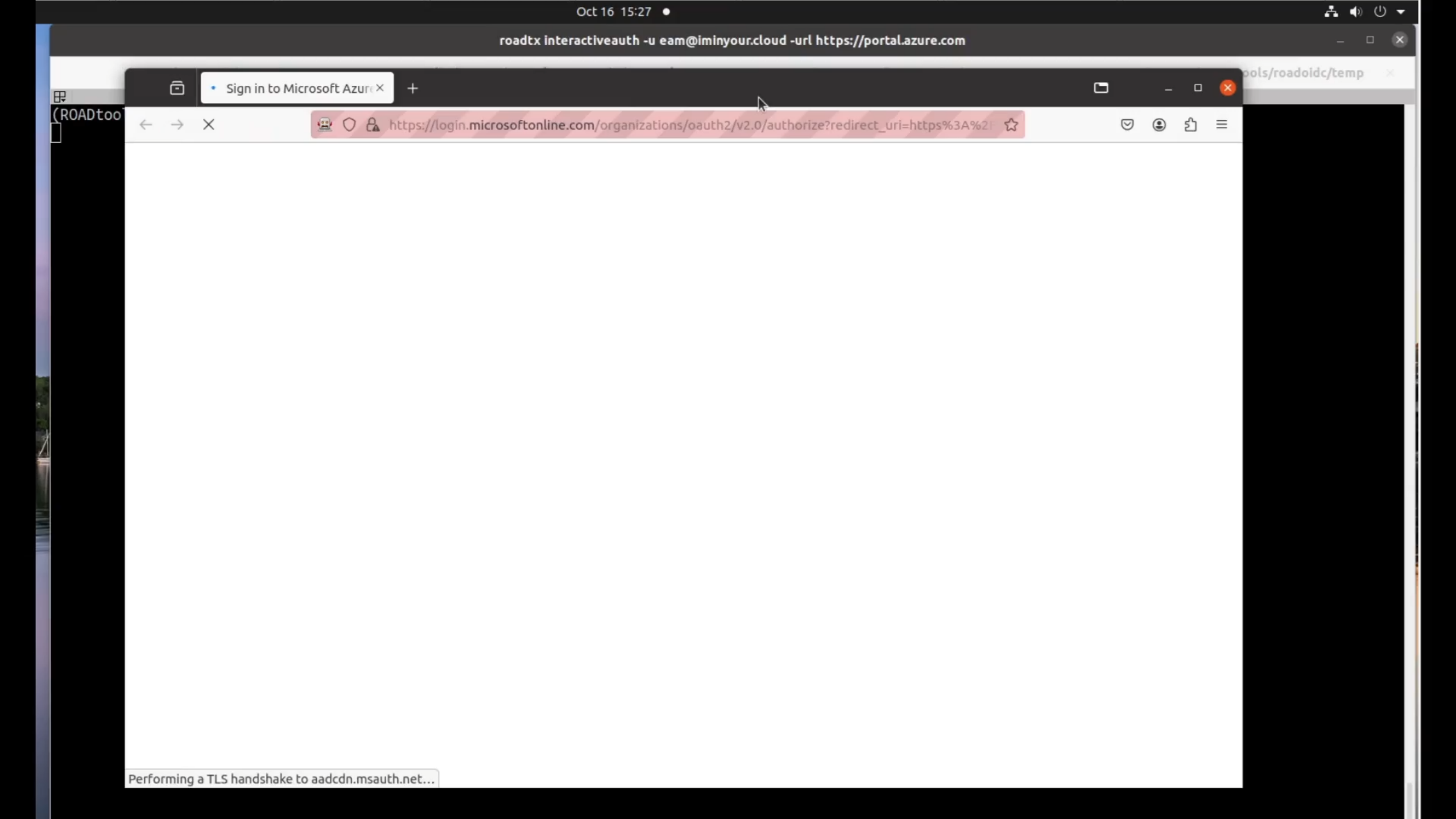

```
{
  "alg": "RS256",
  "kid": "6oEQag-izr3DUuZ3Fwrcq1S1Xbk-mkNxRGX2mIp5kiM",
  "typ": "JWT"
}
```


PAYLOAD: DATA


```
{
  "iss": "https://eu-west.azureauth.duosecurity.com",
  "sub": "ogJGqFhPCFf30ox1RvVNe_R0iWW_k725lYJPeaIoj04",
  "aud":
"YXBpLWEyYTM0NDFkLmR1b3NlY3VyaXR5LmNvbTpESTNVWEc5M0xCQzR
XSk80OExOUQ==",
  "exp": 1749727859,
  "iat": 1749727559,
  "nonce":
"AwABEgEAAAADAOz_BQD0_7nmdPujJwS0nhFHB2JRGFuibnS2erMtZBy
N-
uBXaemX7_7taa1LIL_pBec_Hr7SG5Y0A6oobvoOmO7i2s0KRrkgAA",
  "DuoMfa": "MfaDone",
  "acr": "possessionorinherence",
  "amr": [
    "pop"
  ]
}
```

Creating our own EAM

- Need to implement this protocol in roadoidc
- Instead of checking MFA, immediately redirect back to Entra ID with signed ID token, using roadoidc signing cert
- Does require roadoidc deployment to Azure App Services since this is no longer just based on static flows.


Oct 16 15:27
roadtx interactiveauth -u eam@iminyour.cloud -url https://portal.azure.com
Sign in to Microsoft Azur
https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize?redirect_uri=https%3A%2F
Performing a TLS handshake to aadcdn.msauth.net...

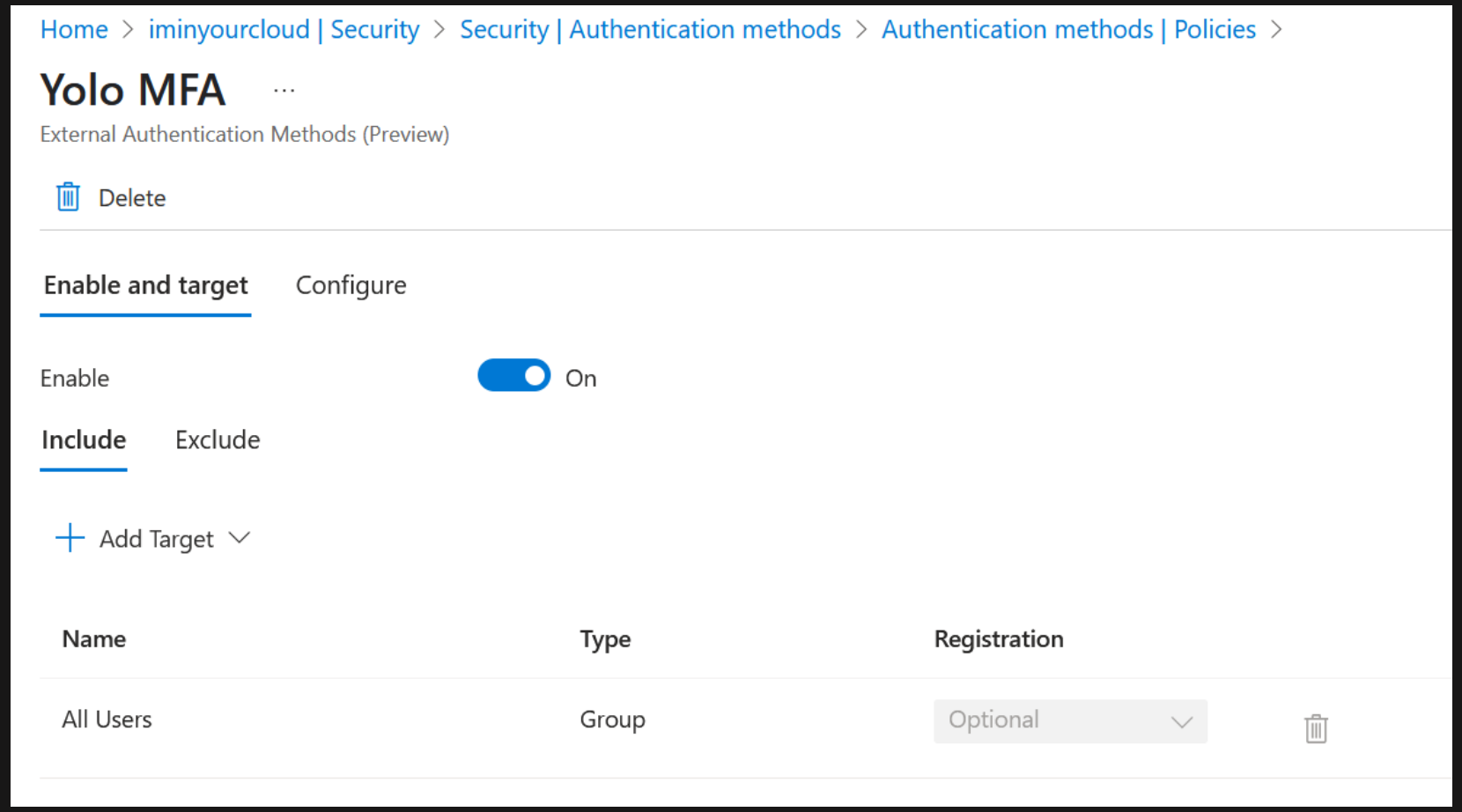
One last hurdle

- We need to have an app in our tenant for the EAM:
  - Must have IdP authorize URL as redirect URL registered
  - Must have openid and profile permissions granted
- Solution:
  - Create app in the tenant and grant consent (requires privileged role)
  - Use existing app with these permissions and replace redirect URL on the fly client side.

Arbitrary MFA for all with EAM

- With roadoidc we can perform fake MFA for any user in scope of the authentication method.
- If we modify the Authentication Methods Policies we can comply with MFA for anyone in scope.
- Does require Global Admin or the MS Graph permission *Policy.ReadWrite.AuthenticationMethod* to configure, so more of a post-exploitation technique.
- EAM does not yet support Auth Strength (even though auth strength is indicated in the protocol)

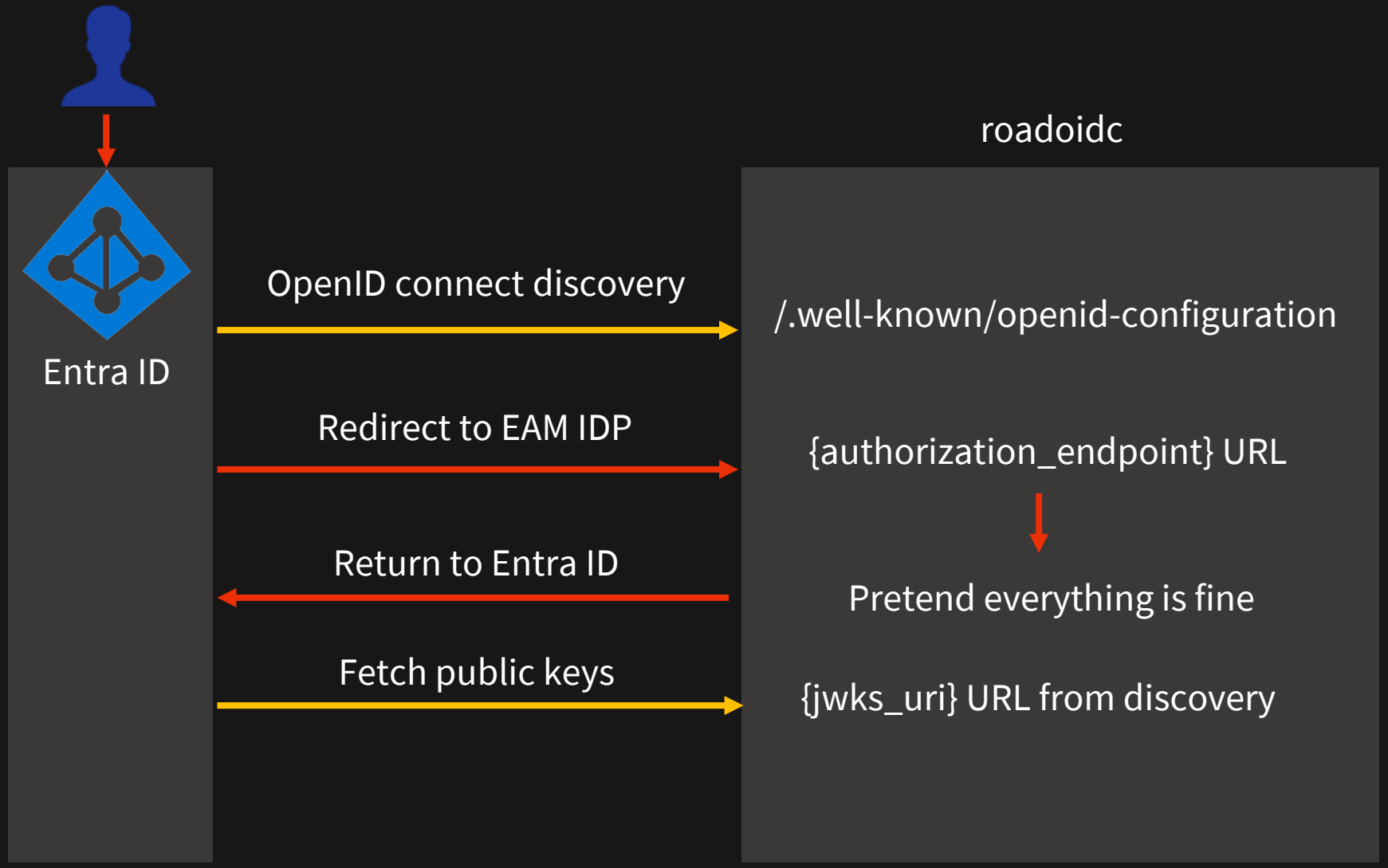
Add EAM method for victims



What if tenant is using EAM?

- If the tenant is using EAM legitimately, we could bypass MFA AD FS style if we can obtain the signing cert and key from the EAM IdP.
- Dumping the cert + key from a third party IdP maybe a bit far fetched.
- What if we can add our own keys?

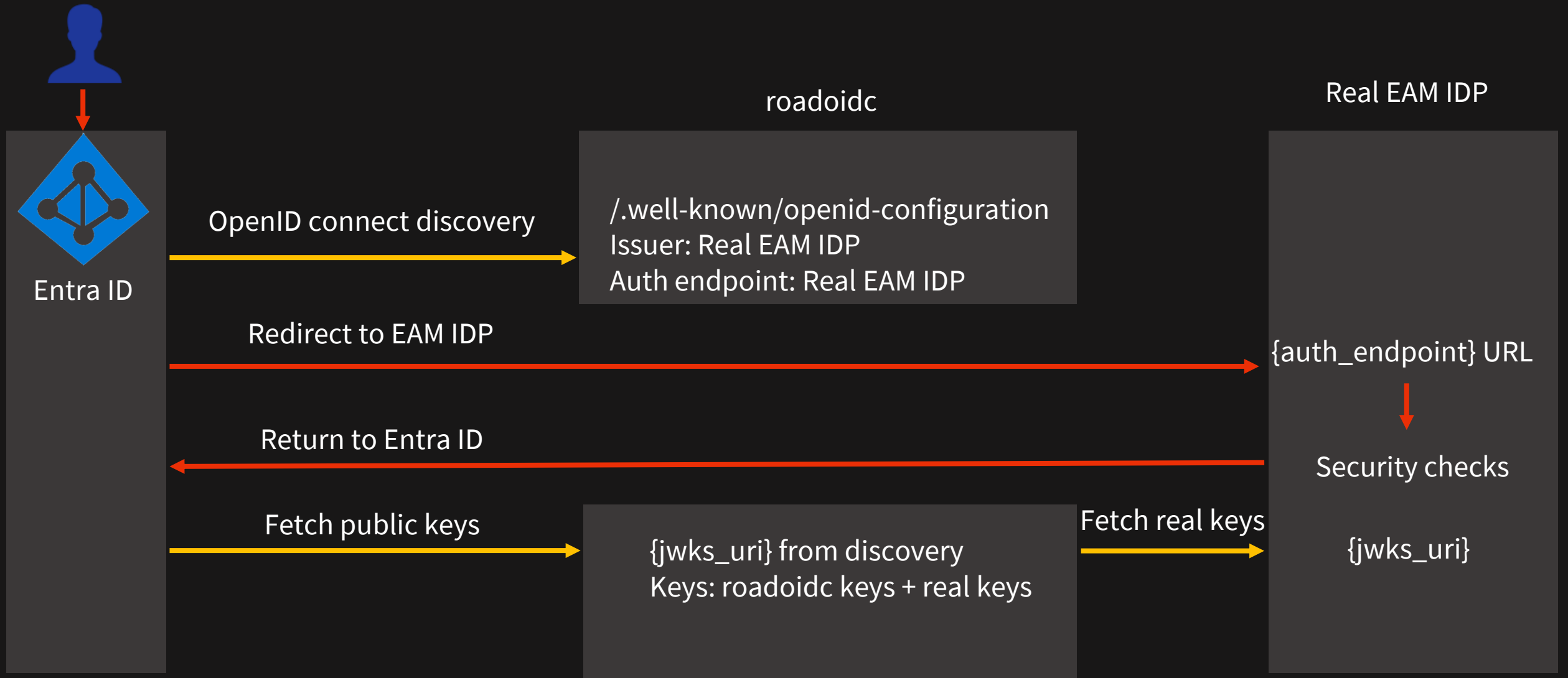
Theoretical OIDC backdoor



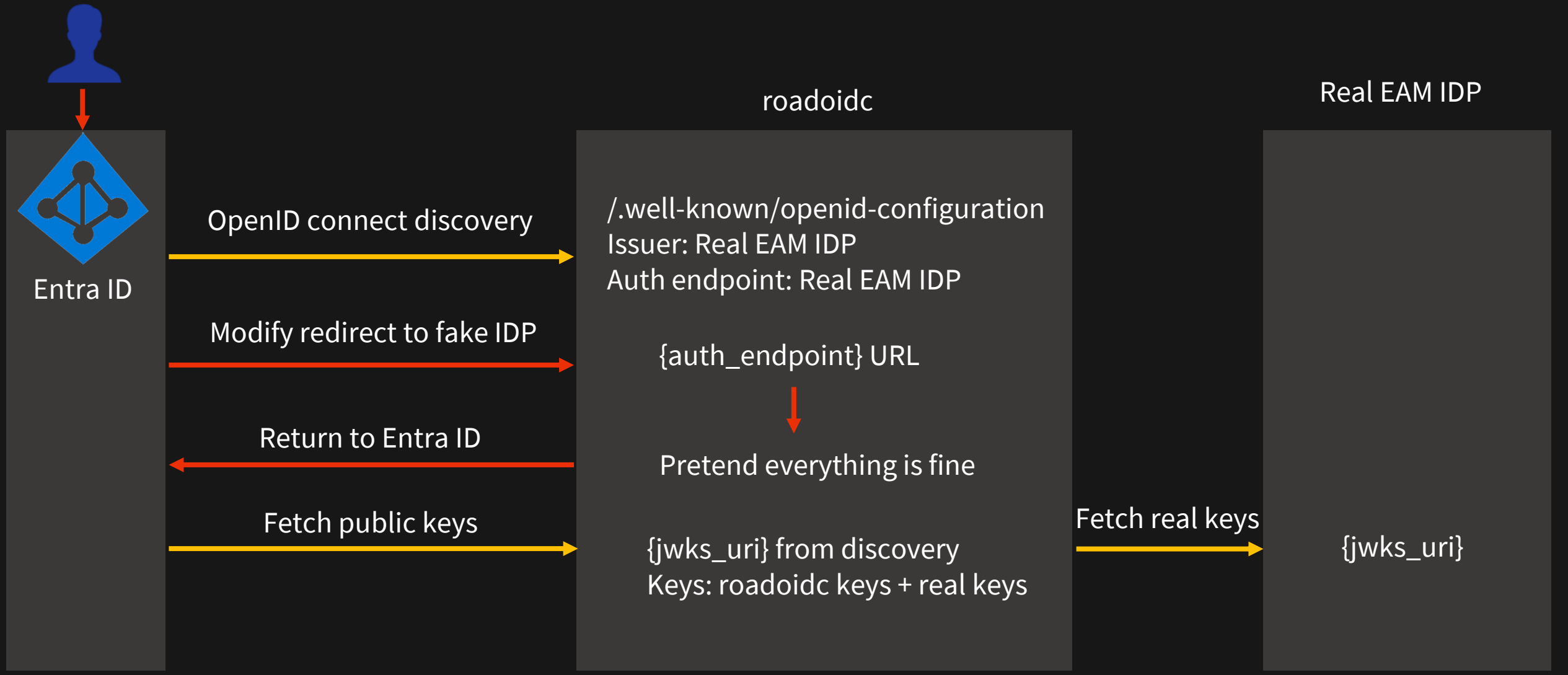
Problems with this backdoor

- We can fake MFA but that would affect the security of the victim tenant.
- Could redirect to the real EAM IdP but then issuer would not match.
- User is redirected to untrusted domain.

Theoretical OIDC backdoor 2



Backdoor 2 – attacker flow



Potential problems with this backdoor

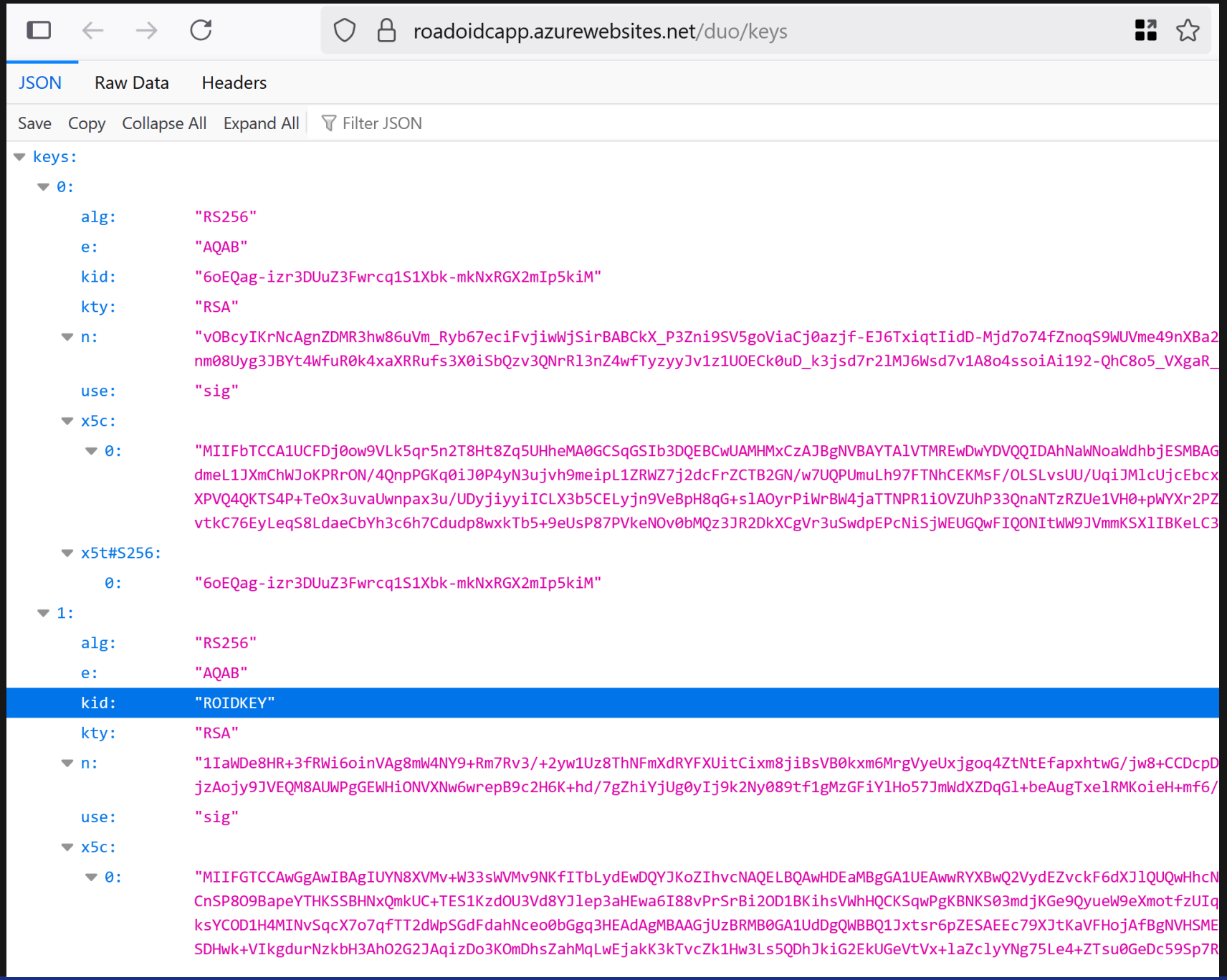
- This is never going to work.
- Why would it be allowed to host the discovery document on a different domain than the issuer.
- Why would it be allowed to host the keys on a different domain than the issuer.

Actual problem with this backdoor

- It does work
- Because Microsoft

In practice


roadoidcapp.azurewebsites.net/duo/.well-known/openid-configuration
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
authorization_endpoint: "https://eu-west.azureauth.duosecurity.com/authorization"
claims_supported: (20)["sub", "iss", "cloud_instance_name", "cloud_instance_host_name", "cloud_graph_host_name", "auth_time", …]
device_authorization_endpoint: "https://login.microsoftonline.com/common/oauth2/v2.0/devicecode"
end_session_endpoint: "https://login.microsoftonline.com/common/oauth2/v2.0/logout"
frontchannel_logout_supported: true
http_logout_supported: true
id_token_signing_alg_values_supported:
0: "RS256"
issuer: "https://eu-west.azureauth.duosecurity.com"
jwks_uri: "https://roadoidcapp.azurewebsites.net/duo/keys"
request_uri_parameter_supported: false


roadoidcapp.azurewebsites.net/duo/keys
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
keys:
0:
alg: "RS256"
e: "AQAB"
kid: "6oEQag-izr3DUuZ3Fwrcq1S1Xbk-mkNxRGX2mIp5kiM"
kty: "RSA"
n: "vOBcyIKrNcAgnZDMR3hw86uVm_Ryb67eciFvjiwWjSirBABCkX_P3Zni9SV5goViaCj0azjf-EJ6TxiqtIidD-Mjd7o74fZnoqS9WUVme49nXBa2
nm08Uyg3JBYt4WfuR0k4xaXRRufs3X0iSbQzv3QNrRl3nZ4wfTyzyyJv1z1UOECk0uD_k3jsd7r2lMJ6Wsd7v1A8o4ssoiAi192-QhC8o5_VXgaR_
use: "sig"
x5c:
0: "MIIFbTCCA1UCFDj0ow9VLk5qr5n2T8Ht8Zq5UHheMA0GCSqGSIb3DQEBCwUAMHMxCzAJBgNVBAYTAlVTMREwDwYDVQQIDAhNaWNoaWdhbjESMBAG
dmeL1JXmChWJoKPRrON/4QnpPGKq0iJ0P4yN3ujvh9meipL1ZRWZ7j2dcFrZCTB2GN/w7UQPUmuLh97FTNhCEKMsF/OLSLvsUU/UqiJMlcUjcEbcx
XPVQ4QKTS4P+TeOx3uvaUwnpax3u/UDyjiyyiICLX3b5CELyjn9VeBpH8qG+slAOyrPiWrBW4jaTTNPR1iOVZUhP33QnaNTzRZUe1VH0+pWYXr2PZ
vtkC76EyLeqS8LdaeCbYh3c6h7Cdudp8wxkTb5+9eUsP87PVkeNOv0bMQz3JR2DkXCgVr3uSwdpEPcNiSjWEUGQwFIQONItWW9JVmmKSXlIBKeLC3
x5t#S256:
0: "6oEQag-izr3DUuZ3Fwrcq1S1Xbk-mkNxRGX2mIp5kiM"
1:
alg: "RS256"
e: "AQAB"
kid: "ROIDKEY"
kty: "RSA"
n: "1IaWDe8HR+3fRWi6oinVAg8mW4NY9+Rm7Rv3/+2yw1Uz8ThNFmXdRYFXUitCixm8jiBsVB0kxm6MrgVyeUxjgoq4ZtNtEfapxhtwG/jw8+CCDcpD
jzAojy9JVEQM8AUWPgGEWHiONVXNw6wrepB9c2H6K+hd/7gZhiYjUg0yIj9k2Ny089tf1gMzGFiYlHo57JmWdXZDqGl+beAugTxelRMKoieH+mf6/
use: "sig"
x5c:
0: "MIIFGTCCAwGgAwIBAgIUYN8XVMv+W33sWVMv9NKfITbLydEwDQYJKoZIhvcNAQELBQAwHDEaMBgGA1UEAwwRYXBwQ2VydEZvckF6dXJlQUQwHhcN
CnSP8O9BapeYTHKSSBHNxQmkUC+TES1KzdOU3Vd8YJlep3aHEwa6I88vPrSrBi2OD1BKihsVWhHQCKSqwPgKBNKS03mdjKGe9QyueW9eXmotfzUIq
ksYCOD1H4MINvSqcX7o7qfTT2dWpSGdFdahNceo0bGgq3HEAdAgMBAAGjUzBRMB0GA1UdDgQWBBQ1Jxtsr6pZESAEEc79XJtKaVFHojAfBgNVHSME
SDHwk+VIkgdurNzkbH3AhO2G2JAqizDo3KOmDhsZahMqLwEjakK3kTvcZk1Hw3Ls5QDhJkiG2EkUGeVtVx+laZclyYNg75Le4+ZTsu0GeDc59Sp7R

Demo

Activities X-terminal-emulator Aug 27 11:17

user@ubuntu:~/ROADtools

user@ubuntu:~/ROADtools 155x43

```
(ROADtools) → ROADtools git:(master) ✗ roadtx -p 127.0.0.1:8082 keepassauth -u duo2@iminyour.cloud
```

Settings

requests and responses passing throug

	Replace
ansport\-Security.*$	X-XSS-Protection: 0
)S.*	
KU.*	
tps://eu-west.azur...	action="https://road

ocket messages passing through the Pr

Configuration library

Only apply to in-scope items

Add	Enabled	Direction	Match	Replace
Edit				

Recap

- Discovery URL we configure:
  - https://roadoidcapp.azurewebsites.net/duo/.well-known/openid-configuration
- Discovery document gives issuer (for DUO):
  - https://eu-west.azureauth.duosecurity.com
- Discovery document keys URL:
  - https://roadoidcapp.azurewebsites.net/duo/keys
  - Gives both the backdoor keys + the real keys fetched from DUO
- Discovery document gives the real authorization page from DUO
  - Real users get redirected to DUO and MFA keeps working as it should
  - Attacker can intercept the redirect and send it to roadoidc to bypass DUO
  - Only Entra ID communicates with roadoidc (so no domain in EDR/FW logs)

Let's read the specs

TOC

4.3. OpenID Provider Configuration Validation

If any of the validation procedures defined in this specification fail, any operations requiring the information that failed to correctly validate MUST be aborted and the information that failed to validate MUST NOT be used.

The `issuer` value returned MUST be identical to the Issuer URL that was used as the prefix to `/.well-known/openid-configuration` to retrieve the configuration information. This MUST also be identical to the `iss` Claim value in ID Tokens issued from this Issuer.

Ref: https://openid.net/specs/openid-connect-discovery-1_0.html#ProviderConfigurationValidation

Let's read Microsoft's own docs

Discovery of provider metadata

An external identity provider needs to provide an OIDC Discovery endpoint. This endpoint is used to get more configuration data. The *full* URL, including *.well-known/oidc-configuration*, must be included in the Discovery URL configured when the EAM is created.

The endpoint returns a Provider Metadata JSON document hosted there. The endpoint must also return the valid content-length header.

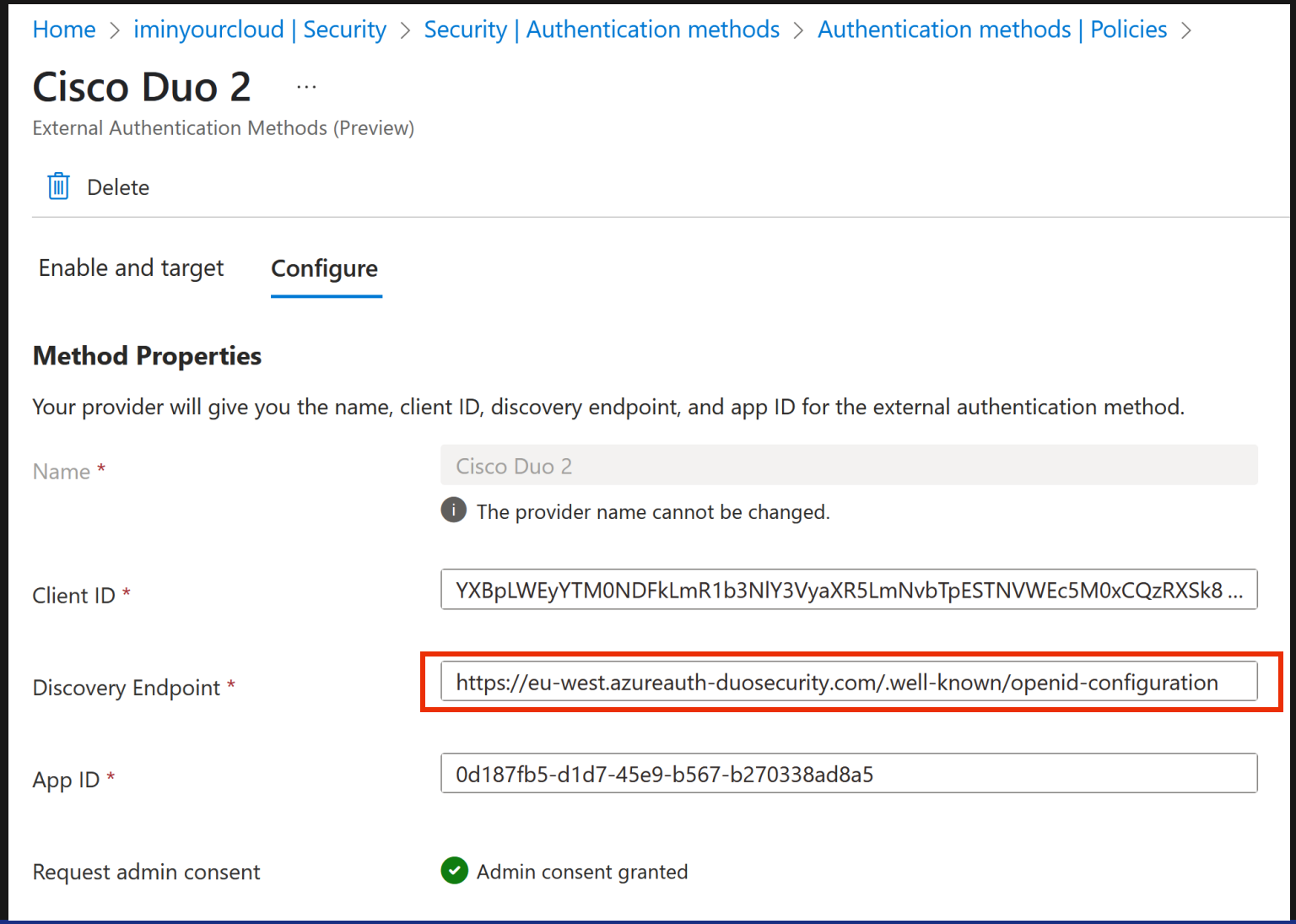
The following table lists the data that should be present in the metadata of the provider. These values are required for this extensibility scenario. The JSON metadata document may contain more information.

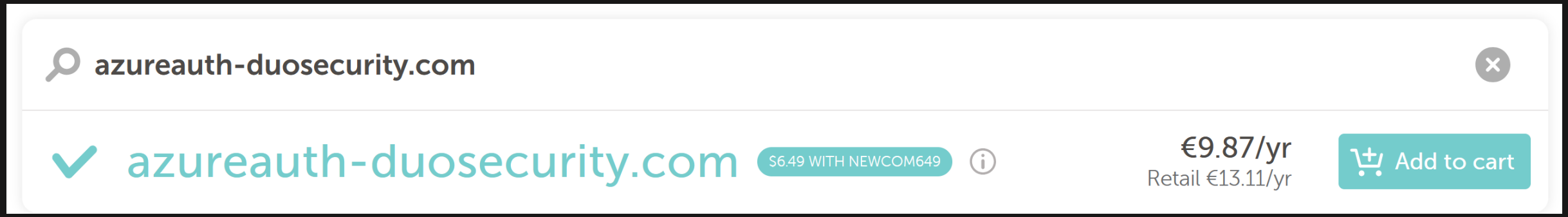
For the OIDC document with the values for Provider Metadata, see Provider Metadata.

Expand table

Metadata value	Value	Comments
Issuer		This URL should match both the host URL used for discovery and the iss claim in the tokens issued by the provider's service.

Real EAM or not?



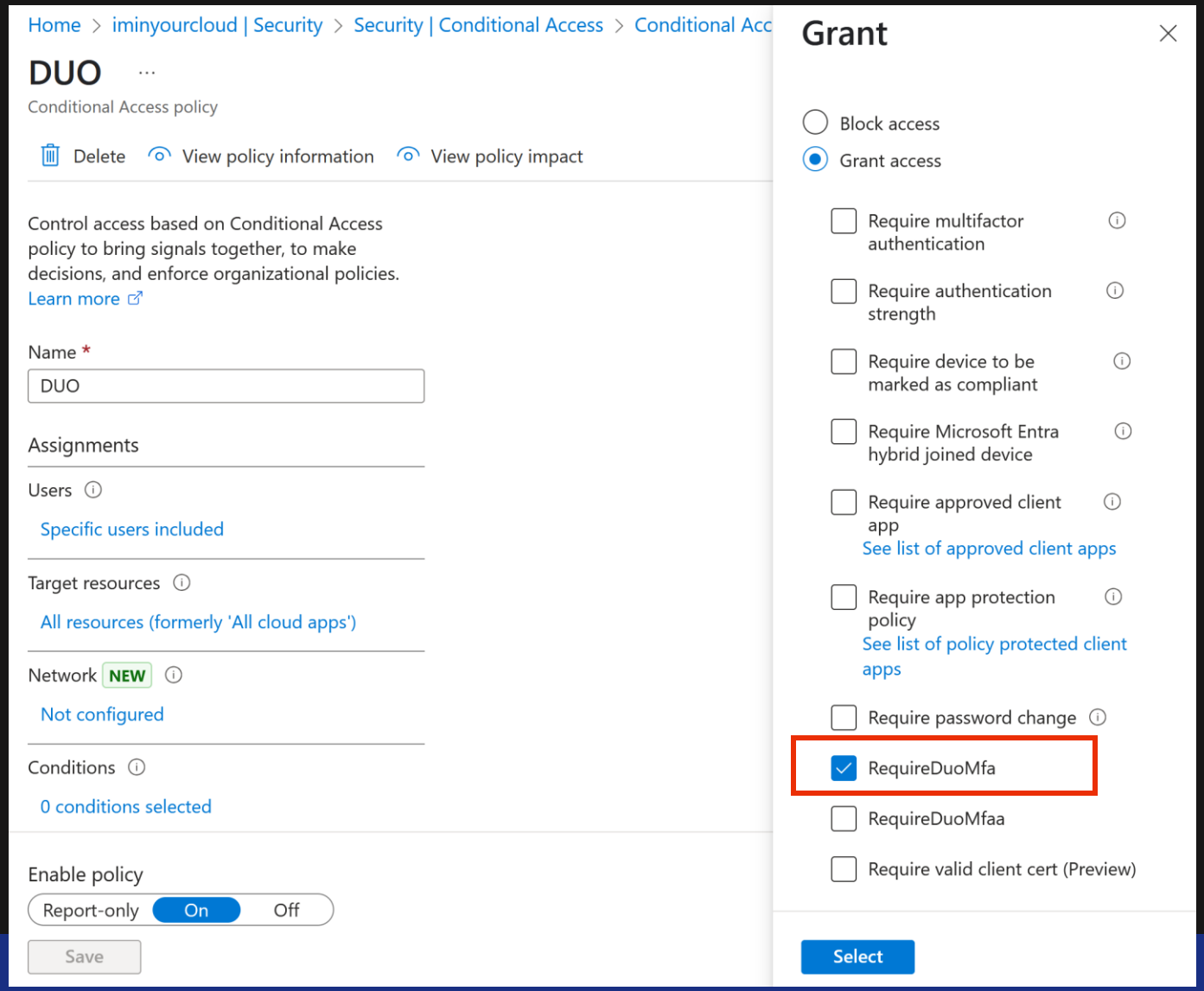

azureauth-duosecurity.com
azureauth-duosecurity.com
$6.49 WITH NEWCOM649
€9.87/yr
Retail €13.11/yr
Add to cart

CA Custom Controls

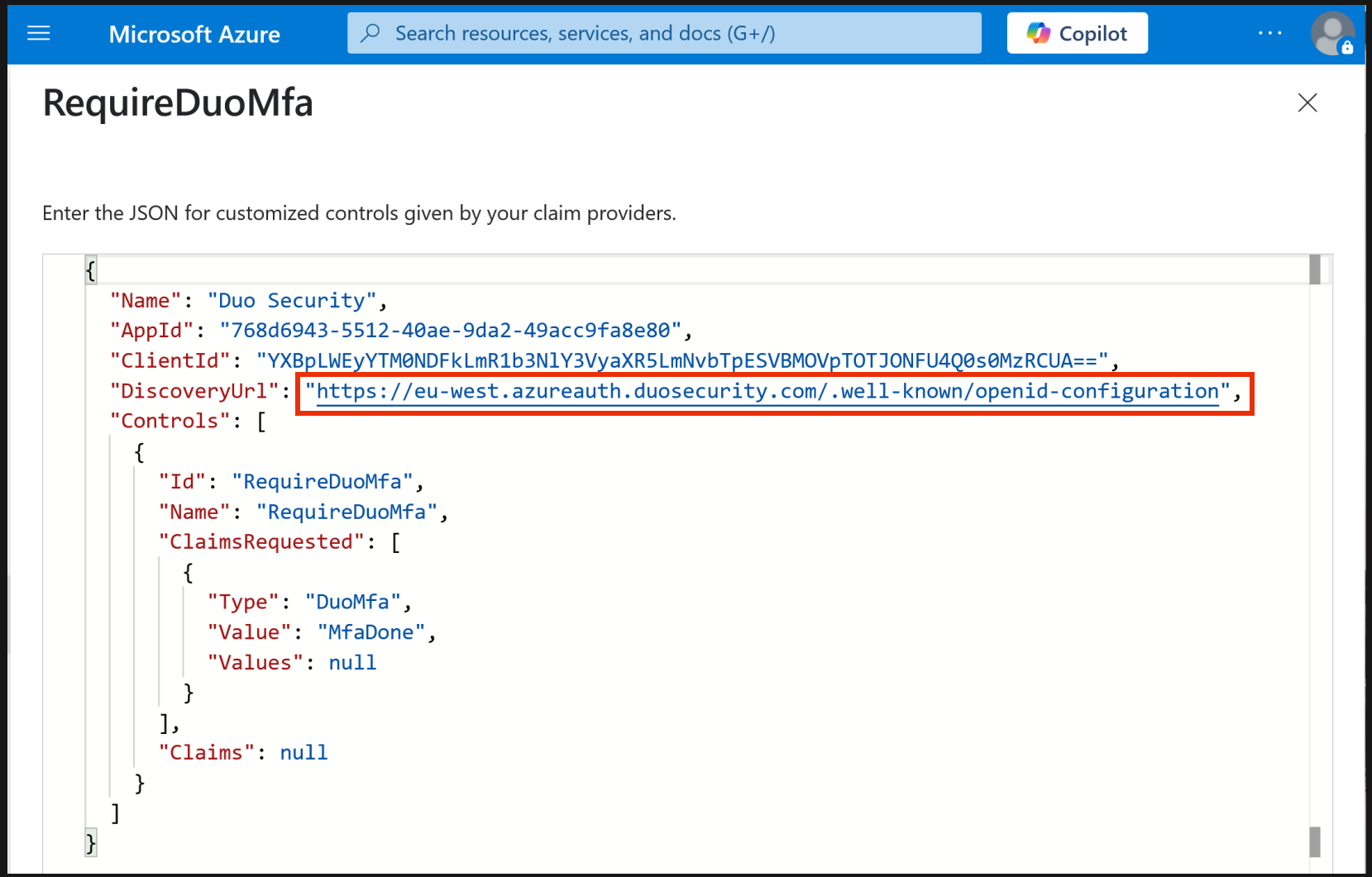
Custom controls in Conditional Access

- Essentially the predecessor of EAM
- Been around for a few year, announced in 2020 that it would not become GA
- Companies still use it
- Until EAM was around the only way to use external MFA providers

Custom controls

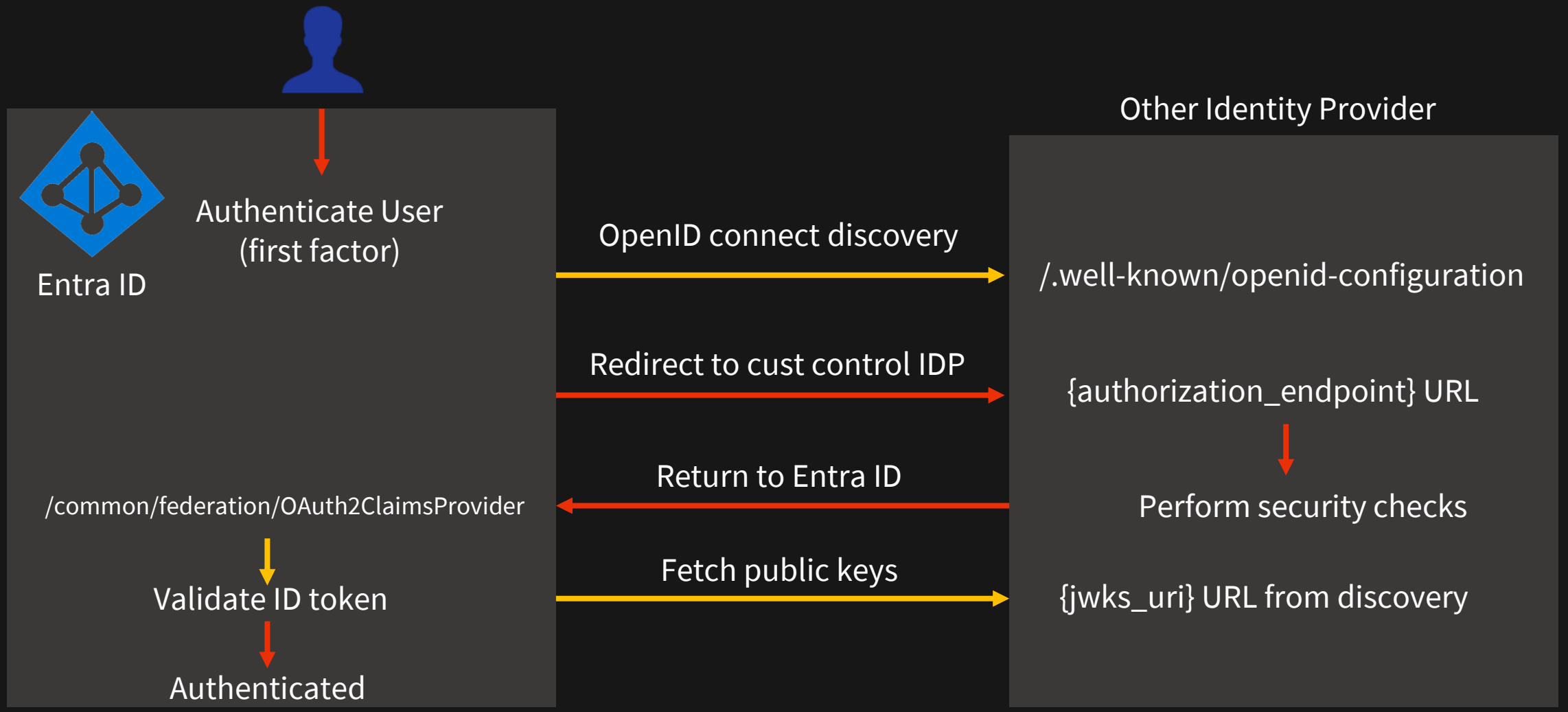


Custom control configuration json



Custom controls and OIDC

Gold = Entra backend request
Red = User flow



Custom controls and OIDC

- Implementation is identical between EAM and Custom Controls.
- Can perform the exact same attacks: modify the Custom Control discovery URL and then inject backdoor keys.
- No need for app registration and redirect URL check in this case.
- Slightly different response is required to make it work.

Disclosure process

Disclosure process

- Sent two reports to MSRC
  - One describing the bug in Custom Controls
  - One describing the bug in EAM
- The bug in EAM was closed as a duplicate of the Custom Controls report
- The bug in Custom Controls was closed as “not a vulnerability” since “admins are free to change the URL at any time and you need privileged access to do this”.
- They also pointed out that Custom Controls will be replaced by EAM which is “better”.

What could be improved

- Implement the OAuth mandatory security checks
- Don’t use the discovery URL pattern but use the issuer and then do the discovery based on the *.well-known/openid-configuration* suffix.
  - This is what is used in federated credentials on apps, which is why this attack doesn’t work there.

Attack flow and detection

Modifying or adding EAM

- Can be done in the Azure portal / Entra Admin portal
- Both will use the Microsoft Graph endpoint */authenticationMethodsPolicy/authenticationMethodConfigurations/*
- Quick detection for policy modifications (assuming you have Graph Activity logs)

```
MicrosoftGraphActivityLogs
| where RequestUri contains "authenticationMethodsPolicy"
| where RequestMethod == "PATCH"
```

- Also recorded in the Entra audit logs, where we can get the actual changes

Detection: EAM modification

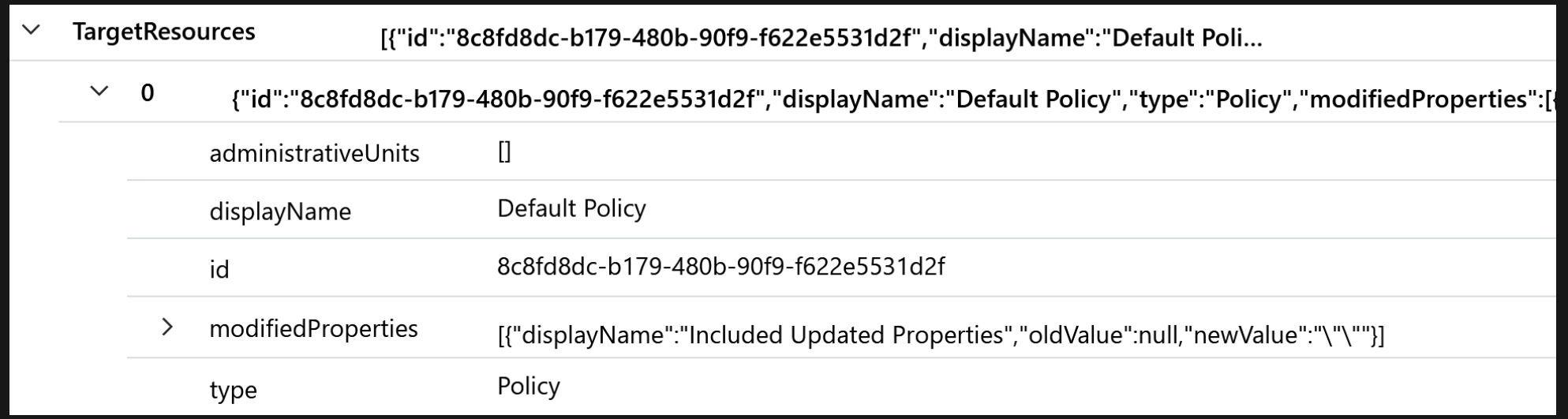
```
AuditLogs
| where OperationName == "Authentication Methods Policy Update"
| extend modifiedProps = TargetResources[0].modifiedProperties
| extend initiatedUser = tostring(InitiatedBy.user.userPrincipalName)
| mv-expand modifiedProps
| extend actualnewvalue = parse_json(tostring(modifiedProps.newValue))
| extend mfa = parse_json(tostring(actualnewvalue)).authenticationMethodConfigurations
| mv-expand mfa
| extend oidc = parse_json(tostring(mfa)).openIdConnectSetting
| where isnotnull(oidc)
| extend discovery = oidc.discoveryUrl
| extend actualoldvalue = parse_json(tostring(modifiedProps.oldValue))
| extend oldmfa = parse_json(tostring(actualoldvalue)).authenticationMethodConfigurations
| mv-expand oldmfa
| extend oldoidc = parse_json(tostring(oldmfa)).openIdConnectSetting
| where isnotnull(oldoidc)
| where tostring(parse_json(tostring(mfa)).id) == tostring(parse_json(tostring(oldmfa)).id)
| extend olddiscovery = oldoidc.discoveryUrl
| where tostring(olddiscovery) !~ tostring(discovery)
| project initiatedUser, olddiscovery, discovery, mfa.displayName

```

Results | Chart

initiatedUser	olddiscovery	discovery	mfa_displayName
dirkjan@iminyour.cloud	https://roadoidcapp.azurewebsites.net/...	https://eu-west.azureauth.duosecurity.com/.well-known/openid-...	Cisco Duo 2
dirkjan@iminyour.cloud	https://roadoidcapp.azurewebsites.net/...	https://eu-west.azureauth.duosecurity.com/.well-known/openid-...	Cisco Duo 2
dirkjan@iminyour.cloud	https://roadoidcapp.azurewebsites.net/...	https://roadoidcapp2.azurewebsites.net/duo/.well-known/openi...	Cisco Duo

Alternative approach

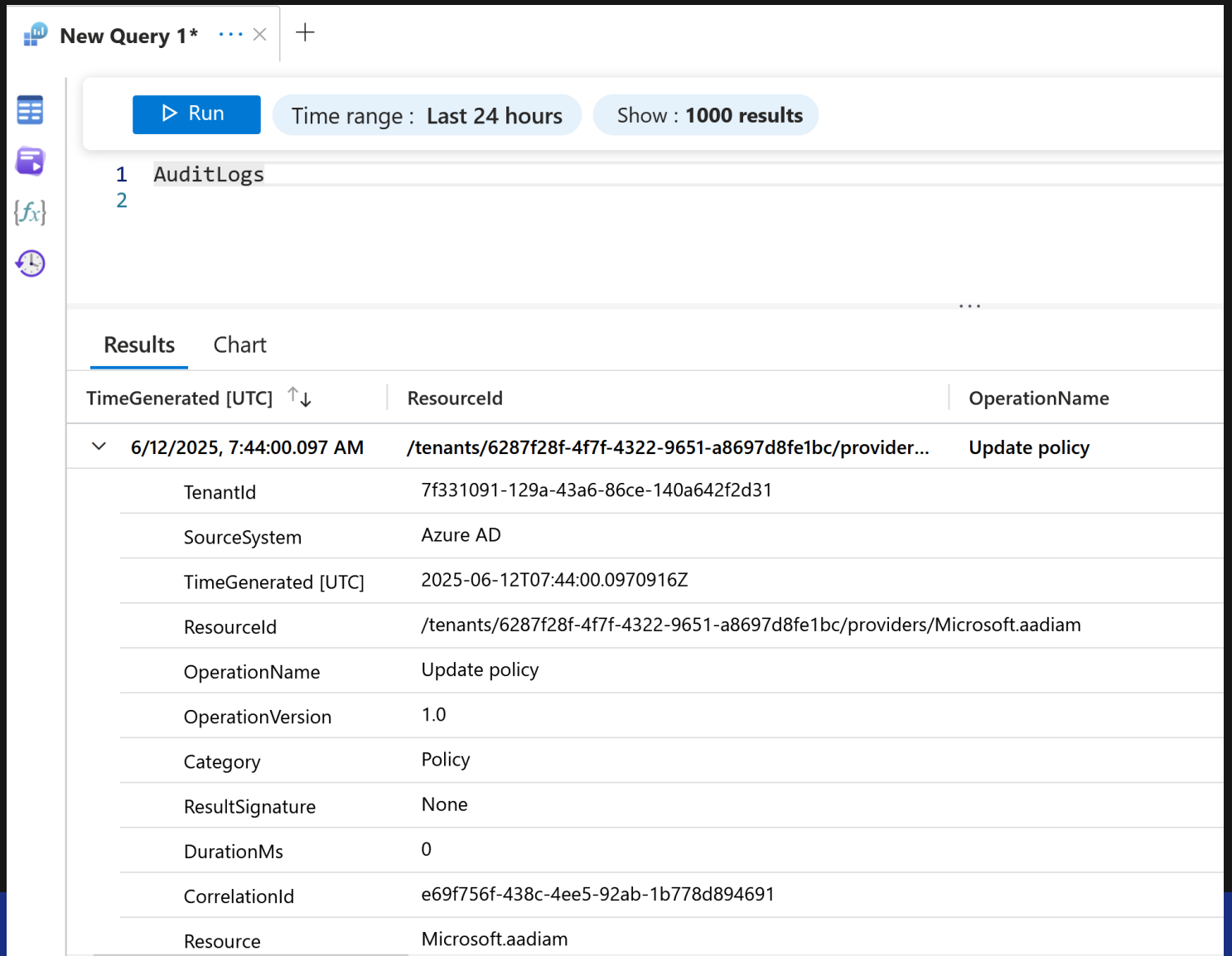
- Use Azure AD Graph “legacy” API https://graph.windows.net
- Modify the “policy” object over an internal API version
- No useful logging ☺ / ☹



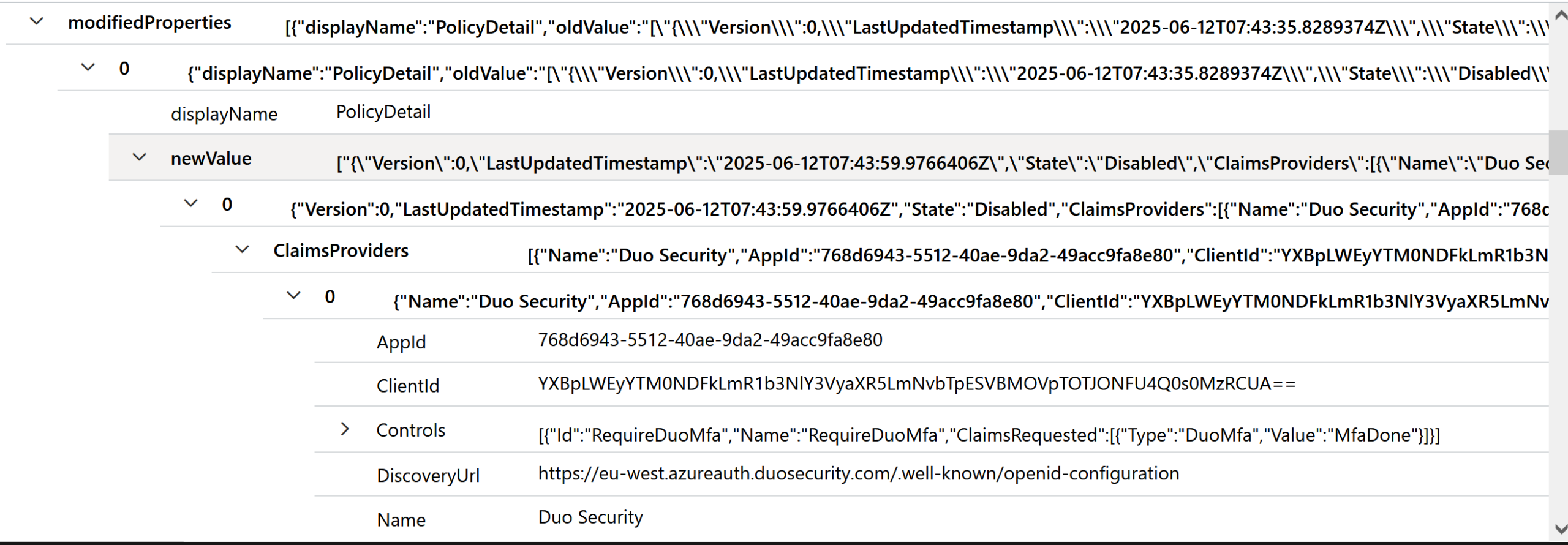
Modifying Custom Controls

- Uses https://main.iam.ad.ext.azure.com/api/ClaimProviders endpoint.
- No “public” API to perform modification.
- Used to not generate any useful logging when I reported it
- Is now recorded in the audit log properly
- Modifying Custom Controls / CA policies via internal API was blocked a few years ago

Custom Control modification



Modified DiscoveryUrl



Run | Time range : **Last 24 hours** | Show : **1000 results**

```
AuditLogs
| where OperationName == "Update policy"
| extend modifiedProps = TargetResources[0].modifiedProperties
| extend initiatedUser = tostring(InitiatedBy.user.userPrincipalName)
| mv-expand modifiedProps
| extend newvalue = modifiedProps.newValue
| mv-expand newvalue
| extend actualnewvalue = parse_json(tostring(newvalue))[0]
| extend claimproviders = parse_json(tostring(actualnewvalue)).ClaimsProviders
| mv-expand claimproviders
| extend newdiscoveryurl = claimproviders.DiscoveryUrl
| extend oldvalue = modifiedProps.oldValue
| mv-expand oldvalue
| extend actualoldvalue = parse_json(tostring(oldvalue))[0]
| extend oldclaimproviders = parse_json(tostring(actualoldvalue)).ClaimsProviders
| mv-expand oldclaimproviders
| extend olddiscoveryurl = oldclaimproviders.DiscoveryUrl
| where tostring(oldclaimproviders.ClientId) == tostring(claimproviders.ClientId)
| where tostring(oldclaimproviders.Name) == tostring(claimproviders.Name)
| where tostring(olddiscoveryurl) !~ tostring(newdiscoveryurl)
| extend clid = oldclaimproviders.ClientId
| extend providername = claimproviders.Name
| project initiatedUser, providername, olddiscoveryurl, newdiscoveryurl
```

Results Chart

initiatedUser	providername	olddiscoveryurl	newdiscoveryurl
dirkjan@iminyour.cloud	Duo Securitya	https://roidapp.azurewebsites.n...	https://roadoidcapp.azurewebsites.net/duo/.well-known/openid-configuration
dirkjan@iminyour.cloud	Duo Securitya	https://roadoidcapp.azurewebsi...	https://roadoidcapp2.azurewebsites.net/duo/.well-known/openid-configuration

Conclusions

Conclusions

- Federated credentials provide new opportunities for taking control and persisting on applications and managed identities – new things to monitor for.
- EAM can be configured as MFA method for a broad scope, helping in post-exploitation scenarios.
- If you actually use EAM or Custom Controls in CA, be on the lookout for “backdoor keys”, which only works because Microsoft refuses to actually implement mandatory OAuth2 security checks.
- New roadoidc release will make this feature available soon.

/OUTSIDER SECURITY

BYO IDP in Entra ID

Persisting and bypassing MFA with OIDC based protocols