SUSE

Bridging Worlds

Linux and Azure Entra ID

Agenda

1. **Introduction**
   Setting the stage (and a quick demo).

2. **Device Enrollment**
   Understanding the process of device enrollment into Entra ID.

3. **Authentication**
   Differences in authentication with an enrolled device.

4. **Windows Hello**
   Authenticating with a TPM PIN.

5. **TGT Retrieval**
   Strategy for TGT retrieval.

6. **Hands-on Experience**
   Demo using the Rust MSAL library..

7. **Q&A**

Please ask questions ***during*** the presentation!

Introduction

Setting the stage

Demo

Demo using the Himmelblau project

Himmelblau: Bridging Azure AD & Intune to Linux

- Interoperability suite for Azure AD and Intune
- Facilitates Linux authentication to Azure AD
- Provides PAM and NSS modules
- Communication to Azure via himmelblaud daemon
- Goal to enforce Intune MDM policies

How does Himmelblau relate to Samba Winbind?

- Himmelblau is a staging ground for Winbind
- Core of Himmelblau is the msal Rust crate
  - Himmelblau is essentially Kanidm integrated with msal
- Given this is the future of Active Directory, Samba belongs in this space!
- Integration of msal into Winbind a work in progress
  - Waiting on Rust/WAF integration
  - Help needed!

How does Himmelblau relate to Ubuntu’s AAD-AUTH?

- AAD-AUTH does not enroll the device
- AAD-AUTH requires explicit client configuration in Azure Entra ID.
  - Himmelblau instead uses the builtin Azure client login (mirroring Windows behavior).
- AAD-AUTH does not perform MFA
  - In fact, it ignores MFA demands from Azure and ***bypasses*** them.
- AAD-AUTH does not use the TPM
- AAD-AUTH relies on Microsoft code
  - Microsoft’s libraries don’t offer MFA, device enrollment, etc.
- AAD-AUTH appears to be abandoned by the developers
  - Last non-bot update was Oct 2023.
- Developers stopped responding to collaboration conversations.

Understanding the Significance

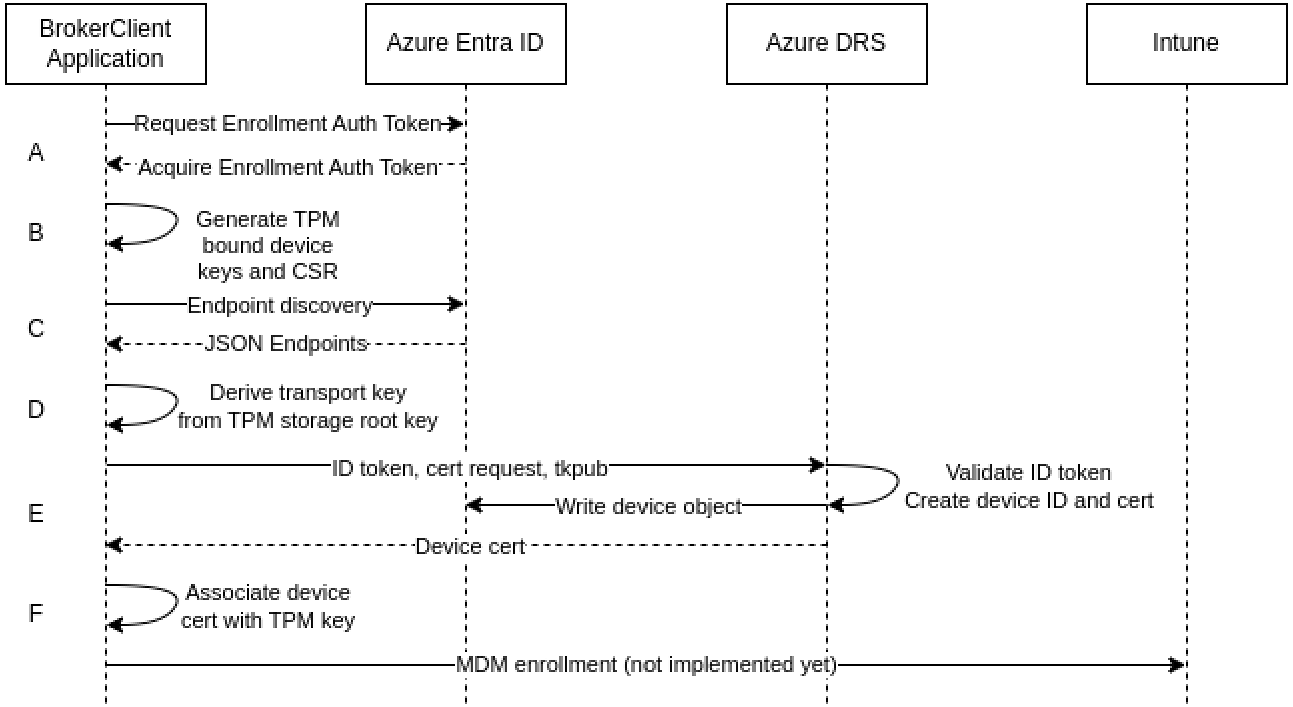
Exploring Microsoft's Proprietary Technologies and Their Impact on Open Source

- Microsoft provides specifications for implementing the protocols.
  - For example: MS-OAPX, MS-OAPXBC, MS-DVRJ, MS-DVRD, MS-KKDCP, MS-KPP, MS-MDE, and MS-MDE2.
  - These are outdated. Microsoft's dochelp has been reluctant to update them.
    - Clearly were written with Azure Entra ID in-mind (refences to AAD througout).
- Microsoft is currently providing ***proprietary Linux binaries*** for their customers to enroll devices.
  - Existing binaries are lacking in integration capabilities, etc.
  - The community would prefer an open solution over installing proprietary code.

Device Enrollment

Understanding the process of device enrollment into Entra ID

Device Enrollment


BrokerClient Application
Azure Entra ID
Azure DRS
Intune
A
Request Enrollment Auth Token
Acquire Enrollment Auth Token
B
Generate TPM bound device keys and CSR
C
Endpoint discovery
JSON Endpoints
D
Derive transport key from TPM storage root key
E
ID token, cert request, tkpub
Validate ID token Create device ID and cert
Write device object
Device cert
F
Associate device cert with TPM key
MDM enrollment (not implemented yet)

Device Enrollment Specification

Written due to a lack of accurate documentation provided by Microsoft regarding Azure Entra ID device enrollment.

- Defines the Device Registration Join Protocol.
- Establishes a secure device identity between physical devices and an Entra ID tenant.
- Based on [MS-DVRJ] and [MS-DVRD].

Authentication Token for Enrollment

- The only requirement for the enrollment token is to request the "https://enrollment.manage.microsoft.com/" resource.

- Supports various authentication types (username/password, refresh token, MFA, etc.) as long as the request is made for the specified resource.

- Follow-up authentications via Windows Hello will include the MFA claim only if the enrollment token used initially included MFA authentication.

Generate RSA Keys and Certificate Signing Request

- Creation of a primary RSA key, known as the Transport Key.
  - The Transport Key facilitates secure communication between devices and Azure services.
- Generation of a secondary RSA key specifically for generating a Certificate Signing Request (CSR).
  - The CSR is used to request a digital certificate from an Azure certificate authority, enabling secure authentication and data transmission.

Enrollment Discovery

A discovery request gets a list of URLs and API parameters essential for Azure API communication

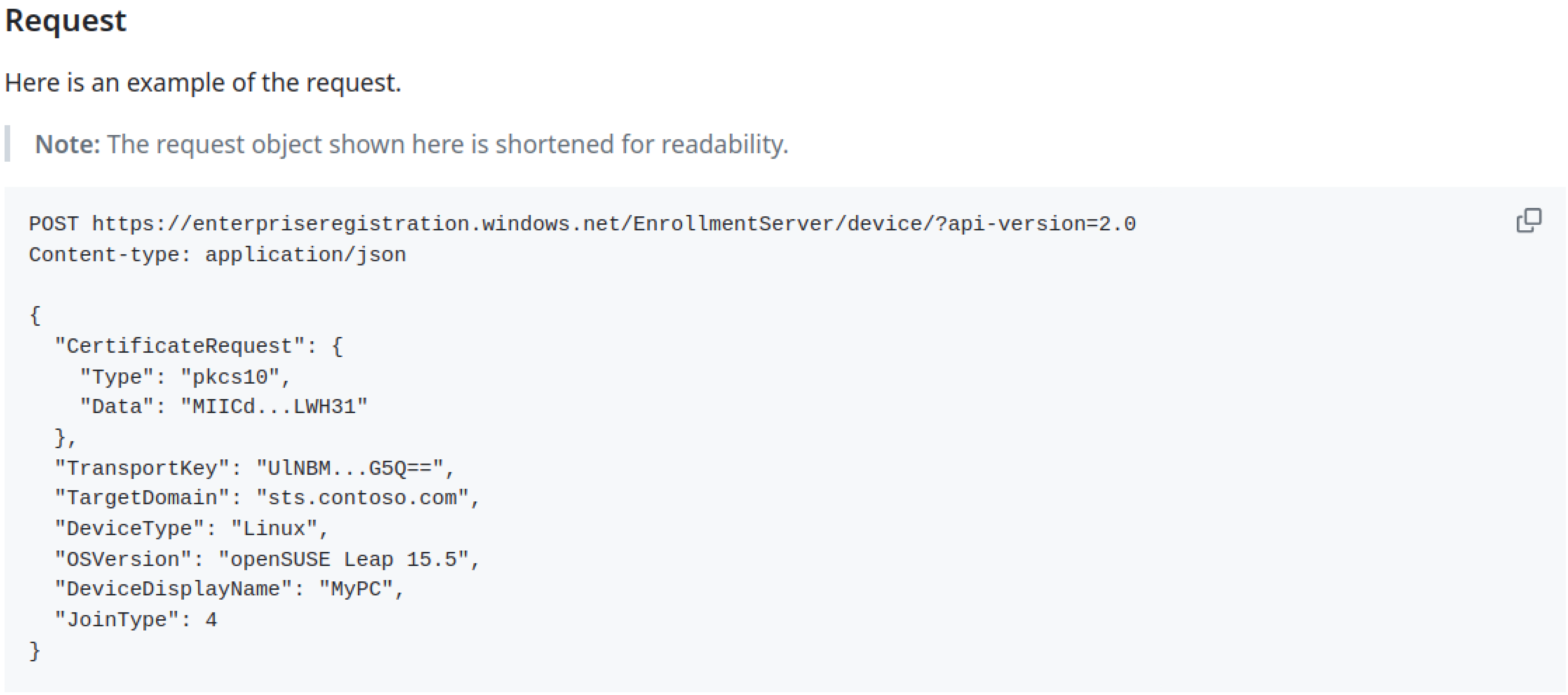
The relevant part of the response for enrollment:

```
"DeviceJoinService": {
    "JoinEndpoint": "https://https://enterpriseregistration.windows.net/EnrollmentServer/device/",
    "JoinResourceId": "urn:ms-drs:enterpriseregistration.windows.net",
    "ServiceVersion": "2.0"
},
```

Device Enrollment

- Sending the Certificate Signing Request (CSR) and public portion of the transport key to Azure for registration, utilizing the discovered API.
- Inclusion of basic device information such as device name, type (e.g., Windows or Linux), and OS version in the enrollment process.

Device Enrollment



Request

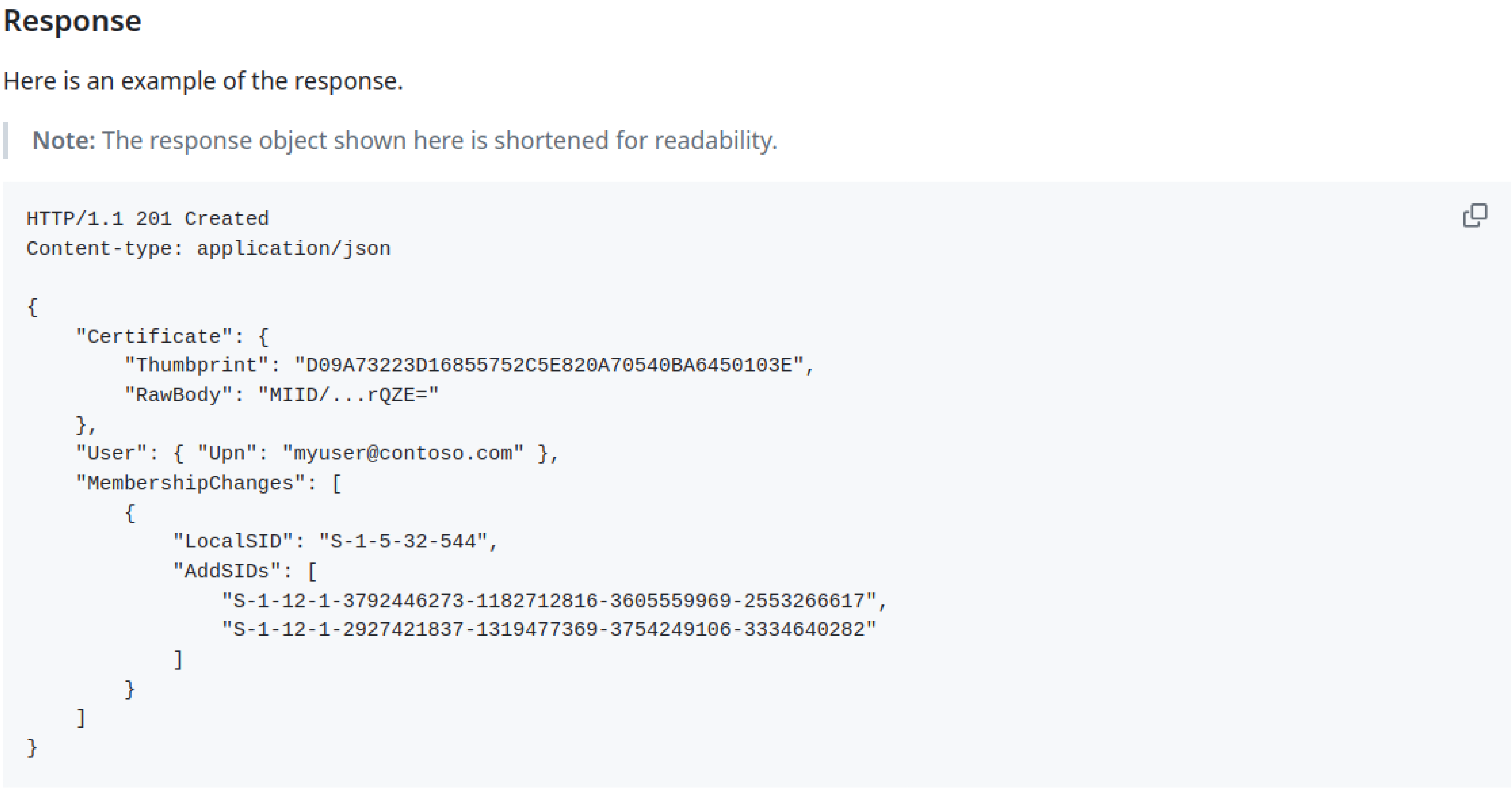
Here is an example of the request.

> **Note:** The request object shown here is shortened for readability.

```
POST https://enterpriseregistration.windows.net/EnrollmentServer/device/?api-version=2.0
Content-type: application/json

{
  "CertificateRequest": {
    "Type": "pkcs10",
    "Data": "MIICd...LWH31"
  },
  "TransportKey": "UlNBM...G5Q==",
  "TargetDomain": "sts.contoso.com",
  "DeviceType": "Linux",
  "OSVersion": "openSUSE Leap 15.5",
  "DeviceDisplayName": "MyPC",
  "JoinType": 4
}
```

Device Enrollment



Response

Here is an example of the response.

> **Note:** The response object shown here is shortened for readability.

```
HTTP/1.1 201 Created
Content-type: application/json

{
    "Certificate": {
        "Thumbprint": "D09A73223D16855752C5E820A70540BA6450103E",
        "RawBody": "MIID/...rQZE="
    },
    "User": { "Upn": "myuser@contoso.com" },
    "MembershipChanges": [
        {
            "LocalSID": "S-1-5-32-544",
            "AddSIDs": [
                "S-1-12-1-3792446273-1182712816-3605559969-2553266617",
                "S-1-12-1-2927421837-1319477369-3754249106-3334640282"
            ]
        }
    ]
}
```

Intune Enrollment

Intune provides device management capabilites – similar to Group Policy in AD

- Currently in progress.
- Windows client enrollment process is understood (defined in [MS-MDE2]).
- Challenges due to significant differences in enrollment endpoints and parameters between operating systems.
  - Linux enrollment utilizes JSON, while Windows utilizes SOAP.

Intune Enrollment (Windows)

Request

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
    xmlns:a="http://www.w3.org/2005/08/addressing"
    xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd"
    xmlns:wst="http://docs.oasis-open.org/ws-sx/ws-trust/200512"
    xmlns:ac="http://schemas.xmlsoap.org/ws/2006/12/authorization">
    <s:Header>
      <a:Action s:mustUnderstand="1">
        http://schemas.microsoft.com/windows/pki/2009/01/enrollment/RST/wstep
      </a:Action>
      <a:MessageID>urn:uuid:0d5a1441-5891-453b-becf-a2e5f6ea3749</a:MessageID>
      <a:ReplyTo>
        <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
      </a:ReplyTo>
      <a:To s:mustUnderstand="1">
        https://enrolltest.contoso.com:443/ENROLLMENTSERVER/DEVICEENROLLMENTWEBSERVICE.SVC
      </a:To>
      <wsse:Security s:mustUnderstand="1">
        <wsse:UsernameToken u:Id="uuid-cc1ccc1f-2fba-4bcf-b063-ffc0cac77917-4">
          <wsse:Username>user@contoso.com</wsse:Username>
          <wsse:Password wsse:Type=
            "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-
1.0#PasswordText">mypassword
          </wsse:Password>
        </wsse:UsernameToken>
      </wsse:Security>
    </s:Header>
```

```
 <s:Body>
   <wst:RequestSecurityToken>
     <wst:TokenType>
 http://schemas.microsoft.com/5.0.0.0/ConfigurationManager/Enrollment/DeviceEnrollmentToken
     </wst:TokenType>
     <wst:RequestType>
       http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue</wst:RequestType>
     <wsse:BinarySecurityToken
       ValueType="http://schemas.microsoft.com/windows/pki/2009/01/enrollment#PKCS10"
       EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd#base64binary">
       DER format PKCS#10 certificate request in Base64 encoding Insterted Here
     </wsse:BinarySecurityToken>
     <ac:AdditionalContext xmlns="http://schemas.xmlsoap.org/ws/2006/12/authorization">
       <ac:ContextItem Name="OSEdition">
         <ac:Value> 4</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="OSVersion">
         <ac:Value>10.0.9999.0</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="DeviceName">
         <ac:Value>MY_WINDOWS_DEVICE</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="MAC">
         <ac:Value>FF:FF:FF:FF:FF:FF</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="MAC">
         <ac:Value>CC:CC:CC:CC:CC:CC</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="IMEI">
         <ac:Value>49015420323756</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="IMEI">
         <ac:Value>30215420323756</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="EnrollmentType">
         <ac:Value>Full</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="DeviceType">
         <ac:Value>CIMClient_Windows</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="ApplicationVersion">
         <ac:Value>10.0.9999.0</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="DeviceID">
         <ac:Value>7BA748C8-703E-4DF2-A74A-92984117346A</ac:Value>
       </ac:ContextItem>
       <ac:ContextItem Name="TargetedUserLoggedIn">
         <ac:Value>True</ac:Value>
       </ac:ContextItem>
     </ac:AdditionalContext>
   </wst:RequestSecurityToken>
 </s:Body>
</s:Envelope>
```

Intune Enrollment (Windows)

Response

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:a="http://www.w3.org/2005/08/addressing"
  xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
   <a:Action s:mustUnderstand="1" >
    http://schemas.microsoft.com/windows/pki/2009/01/enrollment/RSTRC/wstep
   </a:Action>
   <a:RelatesTo>urn:uuid:81a5419a-496b-474f-a627-5cdd33eed8ab</a:RelatesTo>
   <o:Security s:mustUnderstand="1" xmlns:o=
    "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
    <u:Timestamp u:Id="_0">
     <u:Created>2012-08-02T00:32:59.420Z</u:Created>
     <u:Expires>2012-08-02T00:37:59.420Z</u:Expires>
    </u:Timestamp>
   </o:Security>
  </s:Header>
  <s:Body>
   <RequestSecurityTokenResponseCollection
    xmlns="http://docs.oasis-open.org/ws-sx/ws-trust/200512">
    <RequestSecurityTokenResponse>
     <TokenType>
  http://schemas.microsoft.com/5.0.0.0/ConfigurationManager/Enrollment/DeviceEnrollmentToken
     </TokenType>
      <DispositionMessage xmlns="http://schemas.microsoft.com/windows/pki/2009/01/enrollment"/>
      <RequestedSecurityToken>
       <BinarySecurityToken
        ValueType=
 "http://schemas.microsoft.com/5.0.0.0/ConfigurationManager/Enrollment/DeviceEnrollmentProvisionDoc"
        EncodingType=
  "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd#base64binary"
        xmlns=
     "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
        B64EncodedSampleBinarySecurityToken
       </BinarySecurityToken>
      </RequestedSecurityToken>
     <RequestID xmlns="http://schemas.microsoft.com/windows/pki/2009/01/enrollment">0
     </RequestID>
    </RequestSecurityTokenResponse>
   </RequestSecurityTokenResponseCollection>
  </s:Body>
 </s:Envelope>
```

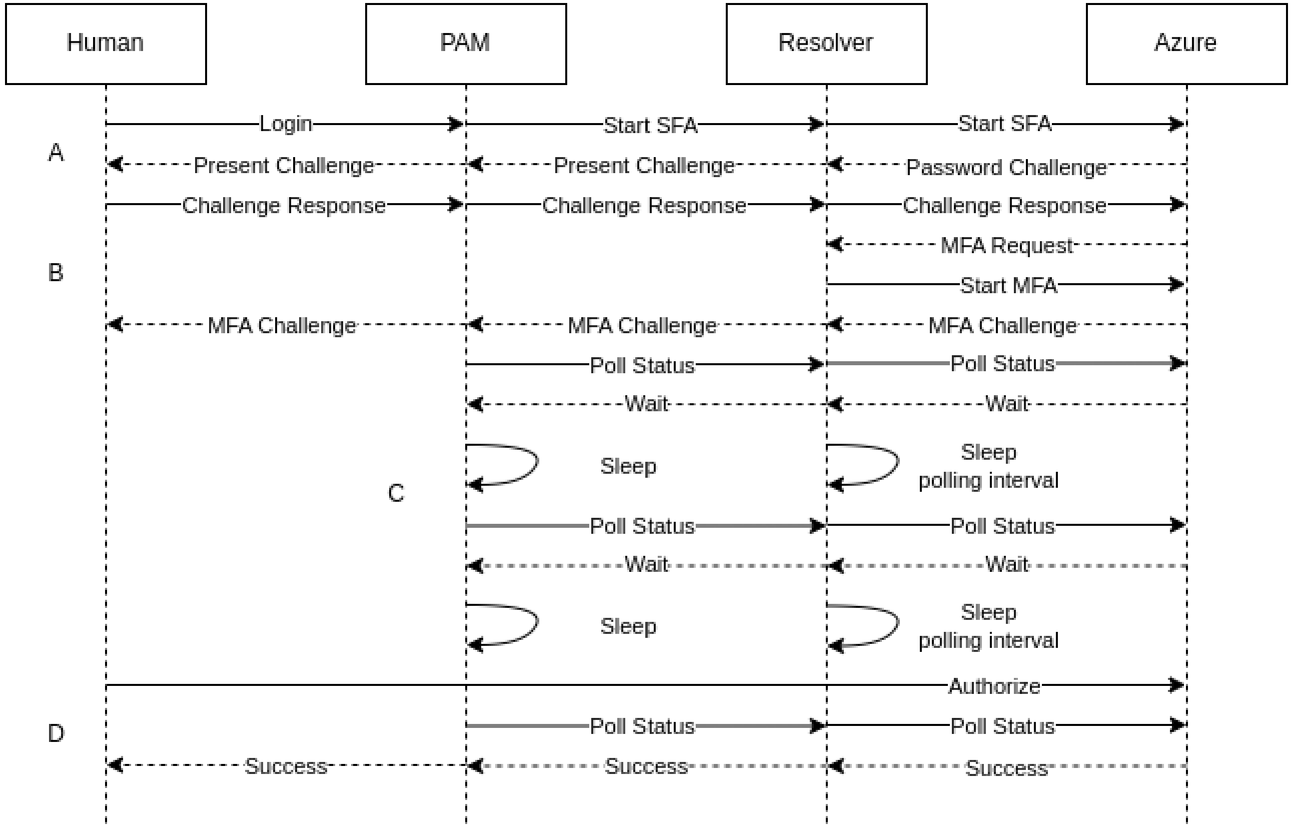
Intune Enrollment (Linux)

- Debug from Linux intune-portal indicates the endpoint: https://fef.amsua0602.manage.microsoft.com/LinuxMDM/LinuxEnrollmentService
  - Endpoint can be found with an authenticated GET to (Microsofts deprecated API): https://graph.microsoft.com/beta/servicePrincipals/appId=0000000a-0000-0000-c000-000000000000/endpoints
- Determined the API expects JSON instead of SOAP.

Authentication

Differences in authentication with
an enrolled device

MFA Authentication for Enrollment



Authentication Differences

- Few differences in how the initial authentication is performed.
- Microsoft expects authentication via a browser window, while Linux performs authentication through PAM prompts.
- Solution: Emulate browser behavior through web requests.
  - Emulating browser behavior poses the risk of potential issues if Microsoft alters the process.
- Alternative approach: Utilize Device Authorization Grant (DAG) for authentication.
  - Drawback is that we can’t force MFA via a DAG, but it will allow it if required for the user.
  - Additional drawback is that DAG can be disabled by the admin.
  - MSAL library supports DAG, enabling authorization to be performed on a different device.
  - Current implementation in Himmelblau includes fallback to DAG if browser emulation fails.

MFA Authentication for Enrollment

- MFA authentication is only required once for enrollment.
- Requesting the "https://enrollment.manage.microsoft.com/" resource during authentication is essential.
- Acquisition of the enrollment token grants specialized privileges.
  - The enrollment token can be exchanged for various services, including a Primary Refresh Token (PRT).
- Sequential exchange of the enrollment token:

1) Exchange for a DRS service token, facilitating actual enrollment.
2) Exchange for an Intune enrollment token.
3) Exchange for a Primary Refresh Token (PRT).
4) Exchange the PRT for a token to enroll a key in Windows Hello.

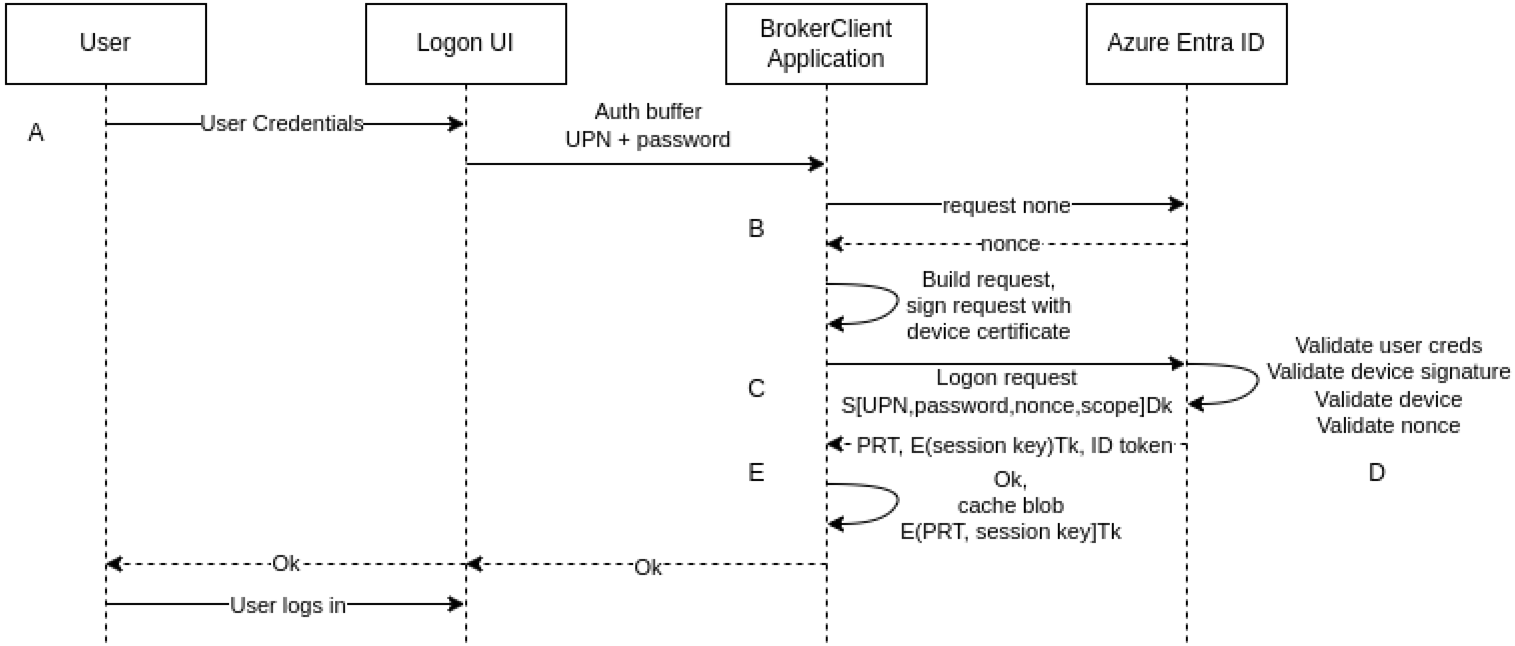
The Hello key can subsequently be unlocked and used for authentication, without additional MFA prompts.

Primary Refresh Token (PRT): Overview

The Primary Refresh Token (PRT) is a master refresh token, allowing users to access resources securely without repeatedly entering credentials.

```
struct PrimaryRefreshToken {
    token_type: String,
    expires_in: String,
    ext_expires_in: String,
    expires_on: String,
    refresh_token: String,
    refresh_token_expires_in: u64,
    session_key_jwe: String,
    id_token: IdToken,
    client_info: ClientInfo,
    device_tenant_id: String,
    tgt_ad: TGT,
    tgt_cloud: TGT,
    kerberos_top_level_names: String,
}
```

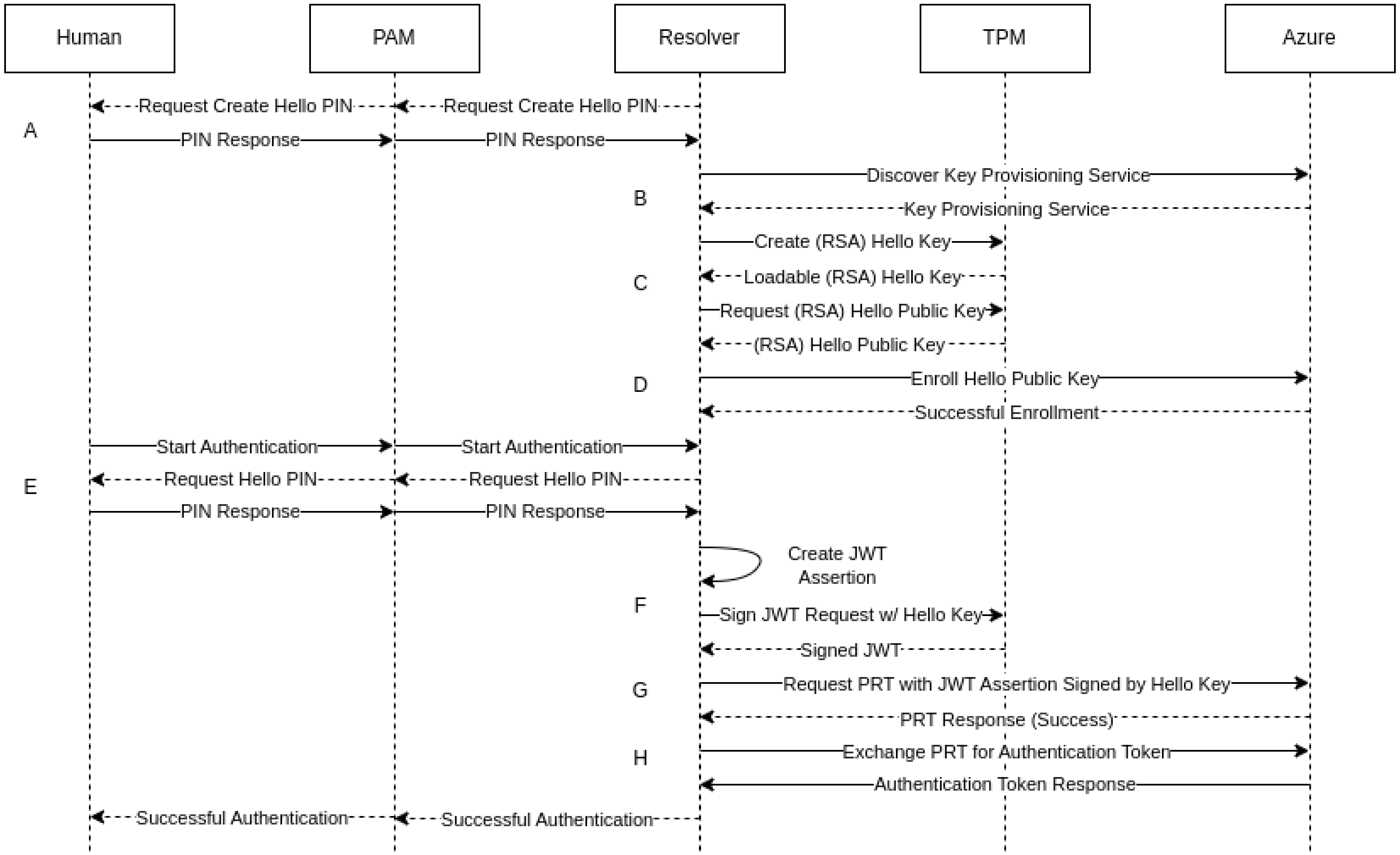
Requesting Primary Refresh Tokens



Windows Hello

Authenticating with a TPM PIN

Windows Hello Enrollment and Authentication



Key Provisioning Discovery

A discovery request gets a list of URLs and API parameters essential for Azure API communication

The relevant part of the response for key provisioning:

```
"KeyProvisioningService": {
    "KeyProvisionEndpoint": "https://enterpriseregistration.windows.net/EnrollmentServer/key/",
    "KeyProvisionResourceId": "urn:ms-drs:enterpriseregistration.windows.net",
    "ServiceVersion": "1.0"
},
```

Creating an RSA TPM Key for Windows Hello

Establishing a Trusted Platform Module (TPM) key on Linux associated with an auth code (referred to as the PIN).

1) Initialize the TPM.
2) Create a Machine Key (the Transport and CSR keys are also nested under this Machine Key).
3) Create an RSA key associated with an auth code (the Hello PIN).
4) The RSA key is stored in the TPM, signing and encryption will be performed by the TPM.

Enrolling a Windows Hello Key

Sending a JSON request to the Key Provisioning service to enroll a Windows Hello key

- Construct a JSON request containing the 'kngc' key.
- The value of 'kngc' is the Windows Hello public key obtained from the TPM, formatted in BcryptRsaKeyBlob format (and base64 encoded).
- Send the JSON request to the Key Provisioning service API, including an access token in the Authorization header.
- If the response is successful, the Windows Hello key has been successfully registered.

Kerberos TGT Retrieval

Strategy for TGT retrieval

TGT Retrieval

A Primary Refresh Token contains on-prem (if configured) and cloud TGTs

- The PRT encapsulates TGTs for both on-prem and cloud environments:
  - On-prem: Represents authentication to local resources.
  - Cloud: Facilitates authentication to cloud-based resources and services within Azure AD.
- Need to implement MS-KKDCP (Kerberos Key Distribution Center Protocol) to leverage these TGTs.

TGTs in the PRT

```
struct TGT {
    clientKey: Option<String>,
    keyType: u32,
    error: Option<String>,
    messageBuffer: Option<String>,
    realm: Option<String>,
    sn: Option<String>,
    cn: Option<String>,
    sessionKeyType: u32,
    accountType: u32,
}
```

Example Cloud TGT

```
struct TGT {
   client_key: Some("eyJhbGciPiJkaXIiLCJlbmWiOiJBMjU2R0NNIiwiY3R4IjoiQ3h4bTdSbfp2WHFqYVZiTXpydENIZlg4YjFHRXJNcDAifQ..y_J4S5WisnPwUcBViV_T8w.SgK3pOIzGVA_l2jV4Ipb21TRzA4Ti48mZY2ZnsE0oeItd320pz8nEv4qTEE5kM2N."),
   key_type: 18,
   error: None,
   message_buffer:
Some("a4IIeTCCCHWgAwIBBaEDAgELox4bHEtFUkJFUk9TLk1JQ1JPU09GVE9OTElORS5DT02kJzAloAMCAQGhHjAcGxp0dXhAMTBmcDd6Lm9ubWljcm9zb2Z0LmNvbaWCBtVhggbRMIIGzaADAgEFoR4bHEtFUkJFUk9TLk1JQ1JPU09GVE9OTElORS5DT0
2iMTAvoAMCAQKhKDAmGwZrcmJ0Z3QbHEtFUkJFUk9TLk1JQ1JPU09GVE9OTElORS5DT02jggZxMIIGbaADAgH/
ooIGZAWCBmAAAFoGAAABAAEgAQAAAOd+iWEmlKdFgHVJWP4eB3w03im4NDOe5ot2CKkpSmFdyRQCfIGq7hPEW3xiU0+gjITiVmpVwiXwiqZPzLXtoQkWI4hz+ccq4V1Jk2WFCYL9ID8zVbpS2Dzqk8UOTEt2PQruX9fwrIgculQMdZoqQNy36vu2f3Se8R1
IuOqEvCpoNd3n/lM+sqrEAzPKAfQz4tAo9aqLFcROCiBLIuD1r6tbibbWgdmgm9C4LRMu29lgTGTiBjOSPxov0UejicFStmv1o44KrAgWqTNPde6q5XZcWb7HyFG+x/Ks/g1T1y1XO1WiP/
lgQHqsSFGHAb5smlUvcsKMIMTouwGYzx6l9XguPievGU1IlgMlVVD8qi6Kn39GGAZrQlsxxO2QG4/aQekbJ3Iz0jsOhdVsimMyAermnpquXFXqEX24Um5worEEK0WKMK3SLg2+AzhP2tiictHXSWZ/4CF4INf2iQ17vDD/k3iA+EtLsFU4fpI/jSFxGcEVc+Cym/
yYP1HaJHGyRHwKVYwXD/HKCEZJC0lK1MVtWcVXJDwvjTtG7RB0xvflrd26GFE2o0Ozg7AlPlD+y1GW6mcjRgOoS3kSkGANYseYP54/PwqeCkJBhLtA6TcBAukuk2lHiZ4hBJGmh11Jw3qc3Bj1StCqD7uFUDUiK12MOz0IfALxIuBRexPudY/
XfJAgmEkcaXVXbVg6+tTvv7b0ZaLM/AHx57eVIIB0d4myVfSjVpt2a8QYix9LsR7R4ojATVA1FgrQ9eqERSSQFmpdPzkTSqK2MFCt4S6sSj0piwtPhV7nrg1GGghtn5SZ+/
+dOklB55ucNAGaeQHCvu190BiiiS+uIPN4fApETxXvgMty30El3XzadpO8I2HnRu+x6v3kSf2T80c60XpNJaZkSLdxXhF8IayQzgG5Ae012CDmG4gIGq7+HLQRGWGEiQi7dxSAkCeFPzONsSIHy8cB8ZZ4Bwl9Ddf0EG9GO9HnvSLVAYAP2SMzpqpF3PvPEoEY6
7SCL72iGUyvPYpyoOcTCi7zCD5gqJmPQBNG+WZgzdXG/Tp7jKnyEERLkdwhzBCMx4iA4wUsYlxhI3WQ24ITt+w2eB2hBzpnOdbSSvP7qeD/aM3YYv+U09NyxFj2bAOlWvVbxbIfqWyAquGKkml6ZrnA+QwbfyQzhxnkT3jhcHU3V6/z6DH2V4TPxBZKde/
Tv4Wuks0zVq9MGxbK8fqYVE7bujBleppj0WxnGIlcvaVUlbz6JQKGVwNVjvm+qeo+RZEwklIVPU0KDMf3n5QVKJOG5yvrcO7KULCDUjuym7XFoabCffFWu8CLZ7PtITJhS1erE9UpgPC8mtiO3dTTlNAnsLLyIm6eLrg8jbDzg5923kK42q1Bv3Fmywm8JHf9nm
MnGIZ7I0bb0Qv4kjPvuWZJ4suIbgexWoRCMaSRcygs8os3XHVeqjH4S7cIGO9oHe9DKeyh1Duwb8FXcmyjwLhVWYTQNT2aDL2A5xwPy5V23ohYKCoFqy+mp0vjne7K5kYeEnzGEoUdKB+2YjDmc2e01jwbkeO1nyRche33RfKsSjSnHFtd5udgno+qzYzdP
dZx7uT50sb4W/eMStLjzJVM4NvaFuFNC059qFwe6XrvOw7m72ZbPh0FgbkA0j/gu/4OO+I1zpiGmAyznG8SKU4xDOGZri6fJ6pPWx7r+PL0gI5gO5K3AvjhLBuMd4+O9V8bHgCJz236NDH7sh+VqlT46eU7Aej4DXs/
OF5nBsaj7sxeWvAhXAirfrRLFdNzsHh0hgyC1Dk0nzWbYLyfaJqjzvSzeQyYSMkfHvaGS1JvTKpkoaJlCx72Fs6LjfPuwQqYpekPp6+wioM6aIKrJUEI1fXeuXfPyoqZARuxHTYDrZTaIXIF/
hDjmETxshIwRu6nxH0SyRhPrHrfALXbkZwYlxMU0V1W0XqI+dJG9UViMBqGZheaMBgbrM3/F04yJJO92+CX2td0nBNluBiL04hbxYBMzVSh0M636fvfcY2IMUYYIO+Ir3jHnI8n8Vm9e2MnWM0Z8GvnNPfPNGRgy/
tr5irVv8F12bZBm+5RsKSK1PB4yTyFS9GW0Pw4dh0wUGCXSedGpEMB3obQm9aU2ZjwSjbO5aVO7nqsAlXFQB+jwoYmronTQ8A83+bRIACmggFFMIIBQaADAgESooIBOASCATTagNbVL7TKBqX6eXIJRkPWLQ5IOP4SMBpWGne/
MaaeIrxDgoqZZVeExC0CVFlnqG0h0m5Sks8CzN8+k8SJdN/01weDokiBuLcKXm6OX1/tLo3Ag9MrGykiArtIUgA/5XC4j84oy67SQBKJgHtA/vkcru++H6jIBP8Age/Klv/ohRXmV7cbMlNm3g4mJIVMare0F4hnNimqE/
dD52HGn7x5mqXyuwltBtSTtgw27WSgnf+RewzvgZah3yMdMmsZVZQ5xW9Z/jk1ph1KNchAcs+LaVJchwH+w7K7ghIYFCDlC/+9g9R2n/SH0GJ8q//
omlGAGfaVqpxTeQN3QE6zo4orol8IxFGnTLHiUfdo6hC3ZPkoFDugVR1I+UF2WKnLWHQNbQqZm9ZHy/fIw+dJRK+oWTzawA=="),
   realm: Some("KERBEROS.MICROSOFTONLINE.COM"),
   sn: Some("krbtgt/KERBEROS.MICROSOFTONLINE.COM"),
   cn: Some("tux@10fp7z.onmicrosoft.com"),
   session_key_type: 0,
   account_type: 2
}
```

Message Buffer

```
b'k\x82\x08y0\x82\x08u\xa0\x03\x02\x01\x05\xa1\x03\x02\x01\x0b\xa3\x1e\x1b\x1cKERBEROS.MICROSOFTONLINE.COM\xa4\'0%\xa0\x03\x02\x01\x01\xa1\x1e0\x1c\x1b\x1a tux@10fp7z.onmicrosoft.com\xa5\x82\x06\xd5a\x82\x06\
xd10\x82\x06\xcd\xa0\x03\x02\x01\x05\xa1\x1e\x1b\x1cKERBEROS.MICROSOFTONLINE.COM \xa210/\xa0\x03\x02\x01\x02\xa1(0&\x1b\x06krbtgt\x1b\x1cKERBEROS.MICROSOFTONLINE.COM \xa3\x82\x06q0\x82\x06m\xa0\x03\x02\x01\
xff\xa2\x82\x06d\x05\x82\x06`\x00\x00Z\x06\x00\x00\x01\x00\x01 \x01\x00\x00\x00\xe7~\x89a&\x94\xa7E\x80uIX\xfe\x1e\x07l4\xde)\xb843\x9e\xe6\x8bv\x08\xa9)Ja]\xc9\x14\x02l\x81\xaa\xee\x13\xc4[lbSO\xa0\x8c\x84\
xe2VjU\xc2%\xf0\x8a\xa6O\xcc\xb5\xed\xa1\t\x16#\x88s\xf9\xc7*\xe1]I\x93e\x85\t\x82\xfd ?3U\xbaR\xd8<\xea\x93\xc5\x0eLKv=\n\xee_\xd7\xf0\xac\x88\x1c\xbaT\x0cu\x9a*@\xdc\xb7\xea\xfb\xb6\x7ft\x9e\xf1\x1dH\xb8\
xea\x84\xbc*h5\xdd\xe7\xfeS>\xb2\xaa\xc4\x033\xca\x01\xf43\xe2\xd0(\xf5\xaa\x8b\x15\xc4N\n K"\xe0\xf5\xaf\xab[\x89\xb6\xd6\x81\xd9\xa0\x9b\xd0\xb8-\x13.\xdb\xd9`Ld\xe2\x063\x92?\x1a/\xd1G\xa3\x89\xc1R\xb6k\
xf5\xa3\x8e\n\xac\x08\x16\xa93Ou\xee\xaa\xe5v\\Y\xbe\xc7\xc8Q\xbe\xc7\xf2\xac\xfe\rS\xd7-W;U\xa2?\xf2 @z\xacHQ\x87\x01\xbeI\x9aU/r\xc2\x8c \xc4\xe8\xbb\x01\x98\xcf\x1e\xa5\xf5x.>\'\xaf\x19MH\x96\x03%UP\xfc\
xaa.\x8a\x9f\x7fF\x18\x06kB[1\xc4\xed\x90\x1b\x8f\xdaA\xe9\x1b\'r3\xd2;\x0e\x85\xd5I\x8ac2\x01\xea\xe6\x9e\x9a\xae\\U\xea\x11}\xb8Rnp\xa2\xb1\x04+E\x8a0\xad\xd2.\r\xbe\x038O\xda\xd8\xa2r\xd1\xd7If\x7f\xe0!x \
xd7\xf6\x89\r{\xbc0\xff\x93x\x80\xf8KK\xb0U8~\x92?\x8d!q\x19\xc1\x15s\xe0\xb2\x9b\xfc\x98?Q\xda$q\xb2D|\nU\x8c\x17\x0f\xf1\xca\x08FI\x0bIJ\xd4\xc5mY\xc5W$</\x8d;F\xed\x10t\xc6\xf7\xe5\xad\xdd\xba\x18Q6\xa3C\
xb3\x83\xb0%>P\xfe\xcbQ\x96\xeag#F\x03\xa8Ky\x12\x90`\rb\xc7\x98?\x9e??\n\x9e\nBA\x84\xbb@\xe97\x01\x02\xe9.\x93iG\x89\x9e!\x04\x91\xa6\x87]I\xc3z\x9c\xdc\x18\xf5J\xd0\xaa\x0f\xbb\x85P5"+]\x8c;=%|\x02\xf1"\
xe0Q{\x13\xeeu\x8f\xd7|\x90 \x98I\x1ciuWmX:\xfa\xd4\xef\xbf\xb6\xf4e\xa2\xcc\xfc\x01\xf1\xe7\xb7\x95 \x80tw\x89\xb2U\xf4\xa3V\x9bvk\xc4\x18\x8b\x1fK\xb1\x1e\xd1\xe2\x88\xc0MP5\x16\n\xd0\xf5\xea\x84E$\x90\x16j]?
9\x13J\xa2\xb60P\xad\xe1.\xacJ=)\x8b\x0bO\x85^\xe7\xae\rF\x1a\x08m\x9f\x94\x99\xfb\xff\x9d:IA\xe7\x9b\x9c4\x01\x9ay\x01\xc2\xbe\xed}\xd0\x18\xa2\x89/\xae \xf3x|\nDO\x15\xef\x80\xcbr\xdfA%\xdd|\xdav\x93\
xbc#a\xe7F\xef\xb1\xea\xfd\xe4I\xfd\x93\xf3G:\xd1zM%\xa6dH\xb7q^\x11\x7c!\xac\x90\xce\x01\xb9\x01\xed5\xd8 \xe6\x1b\x88\x08\x1a\xae\xfe\x1c\xb4\x11\x19a\x84\x89\x08\xbbw\x14\x80\x90\'\x85?3\x8d\xb1"\x07\xcb\
xc7\x01\xf1\x96x\x07\t}\r\xd7\xf4\x10oF;\xd1\xe7\xbd"\xd5\x01\x80\x0f\xd9#3\xa6\xaaE\xdc\xfb\xcf\x12\x81\x18\xeb\xb4\x82/\xbd\xa2\x19L\xaf=\x8ar\xa0\xe7\x13\n.\xf3\x08>`\xa8\x99\x8f@\x13F\xf9f`\xcd\xd5\xc6\xfd:{\
x8c\xa9\xf2\x10DK\x91\xdc!\xcc\x10\x8c\xc7\x88\x80\xe3\x05,b\\a#u\x90\xdb\x82\x13\xb7\xec6x\x1d\xa1\x07:g9\xd6\xd2J\xf3\xfb\xa9\xe0\xffh\xcd\xd8b\xff\x94\xd3\xd3r\xc4X\xf6I\x03\xa5Z\xf5[\xc5\xb9_\xa9I\x80\
xaa\xe1\x8a\x92izf\xb9\xc0\xf9\x0c\x1b\x7f$3\x87\x19\xe4Ox\xe1pu7W\xaf\xf3\xe81\xf6W\x84\xcf\xc4\x16Ju\xef\xd3\xbf\x85\xae\x92\xcd3V\xafL\x1b\x16\xca\xf1\xfa\x98TN\xdb\xba0ez\x9ac\xd1lg\x18\x82\x1c\xbd\xa5T!\
xbc\xfa%\x02\x86W\x03U\x8e\xf9\xbe\xa9\xea>E\x910\x92R\x15=M\n\x0c\xc7\xf7\x9f\x94\x15(\x93\x86\xe7+\xebp\xee\xcaP\xb0\x83R;\xb2\x9b\xb5\xc5\xa1\xa6\xc2}\xf1V\xbb\xc0\x8bg\xb3\xed!2aKW\xab\x13\xd5)\x80\
xf0\xbc\x9a\xd8\x8e\xdd\xd4\xd3\x94\xd0\'\xb0\xb2\xf2"n\x9e.\xb8<\x8d\xb0\xf3\x83\x9fv\xdeB\xb8\xda\xadA\xbfqf\xcb\t\xbc$w\xfd\x9ec\'\x18\x86{#F\xdb\xd1\x0b\xf8\x923\xef\xb9fI\xe2\xcb\x88n\x07\xb1Z\x84B1\
xa4\x91s(,\xf2\x8b7\\u^\xaa1\xf8K\xb7\x08\x18\xefh\x1d\xefC)\xec\xa1\xd4;\xb0o\xc1WrI\xa3\xc0\xb8UY\x84\xd05=\x9a\x0c\xbd\x80\xe7\x1c\x0f\xcb\x95v\xde\x88X(*\x05\xab/\xa6\xa7K\xe3\x9d\xee\xca\xe6F\x1e\x12I\
xc6\x12\x85\x1d(\x1f\xb6b0\xe6sg\xb4\xd6<\x1b\x91\xe3\xb5\x9f$\\\x85\xed\xf7E\xf2\xacJ4\xa7\x1c[]\xe6\xe7`\x9e\x8f\xaa\xcd\x8c\xdd=\xd6q\xee\xe4\xf9\xd2\xc6\xf8[\xf7\x8cJ\xd2\xe3\xcc\x95L\xe0\xdb\xda\x16\
xe1M\x0bN}\xa8\\\x1e\xe9z\xef;\x0e\xe6\xeff[>\x1d\x05\x81\xb9\x00\xd2?\xe0\xbb\xfe\x0e;\xe25\xce\x98\x86\x98\x0c\xb3\x9co\x12)N1\x0c\xe1\x99\xae.\x9f\'\xaaO[\x1e\xeb\xf8\xf2\xf4\x82^`;\x92\xb7\x02\xf8\xe1,\x1b\
x8cw\x8f\x8e\xf5_\x1b\x1e\x00\x89\xcfm\xfa41\xfb\xb2\x1f\x95\xaaT\xf8\xe9\xe5;\x01\xe8\xf8\r{?8^g\x06\xc6\xa3\xee\xcc^Z\xf0!\\\x08\xab~\xb4K\x15\xd3s\xb0xt\x86\x0c\x82\xd494\x9f5\x9b`\xbc\x9fh\x9a\xa3\xce\
xf4\xb3y\x0c\x98H\xc9\x1f\x1e\xf6\x86KRoL\xaad\xa1\xa2e\x0b\x1e\xf6\x16\xce\x8b\x8d\xf3\xee\xc1\n\x98\xa5\xe9\x0f\xa7\xaf\xb0\x8a\x83:h\x82\xab%A\x08\xd5\xf5\xde\xb9w\xcf\xca\x8a\x99\x01\x1b\xb1\x1d6\x03\
xad\x94\xda!r\x05\xfe\x10\xe3\x98D\xf1\xb2\x19pF\xee\xa7\xc4}\x12\xc9\x18O\xacz\xdf\x00\xb5\xdb\x91\x9c\x18#\x13\x14\xd1]V\xd1z\xa5\xf9\xd2F\xf5Eb0\x1a\x86f\x17\x9a0\x18\x1b\xac\xcd\xff\x17N2$\x93\xbd\xdb\xe0\
x97\xda\xd7t\x9c\x13e\xb8\x18\x8b\xd3\x88[\xc5\x80L\xcdT\xa1\xd0\xce\xb7\xe9\xfb\xdfq\x8d\x881F\x18 \xef\xa5\xafx\xc7\x9c\x8f\'\xf1Y\xbd{c\'X\xcd\x19\xf0k\xe74\xf7\xcf4d`\xcb\xfbk\xe6*\xd5\xbf\xc1u\xd9\xb6A\x9b\
xeeQ\xb0\xa4\x8a\xd4\xf0x\xc9<\x85K\xd1\x96\xd0\xfc8v\x1d0P`\x97I\xe7F\xa4C\x01\xde\x86\xd0\x9b\xd6\x94\xd9\x98\xf0J6\xce\xe5\xa5N\xeez\xac\x02U\xc5@\x1f\xa3\xc2\x86&\xae\x89\xd3C\xc0<\xdf\xe6\xd1 \x00\
xa6\x82\x01E0\x82\x01A\xa0\x03\x02\x01\x12\xa2\x82\x018\x04\x82\x014\xda\x80\xd6\xd5/\xb4\xca\x06\xa5\xfayr\tFC\xd6-\x0eH8\xfe\x120\x1aV\x1aw\xbf1\xa6\x9e"\xbcC\x82\x8a\x99eW\x84\xc4-\x02TYg\xa8m!\xd2nR\
x92\xcf\x02\xcc\xdf>\x93\xc4\x89t\xdf\xf4\xd7\x07\x83\xa2H\x81\xb8\xb7\n^n\x8e__\xed.\x8d\xc0\x83\xd3+\x1b)"\x02\xbbHR\x00?\xe5p\xb8\x8f\xce(\xcb\xae\xd2@\x12\x89\x80{@\xfe\xf9\x1c\xae\xef\xbe\x1f\xa8\xc8\
x04\xff\x00\x81\xef\xca\x96\xff\xe8\x85\x15\xe6W\xb7\x1b2Sf\xde\x0e&&ULj\xb7\xb4\x17\x88g6)\xaa\x13\xf7C\xe7a\xc6\x9f\xbcy\x9a\xa5\xf2\xbb\tm\x06\xd4\x93\xb6\x0c6\xedd\xa0\x9d\xff\x91{\x0c\xef\x81\x96\xa1\
xdf#\x1d2k\x19U\x949\xc5oY\xfe95\xa6\x1dJ5\xc8@r\xcf\x8biR\\\x87\x01\xfe\xc3\xb2\xbb\x82\x19X\x14 \xe5\x0b\xff\xbd\x83\xd4v\x9f\xf4\x87\xd0b|\xab\xff\xe8\x9aQ\x80\x19\xf6\x95\xaa\x9cSy\x03w@N\xb3\xa3\x8a+\
xa2_\x08\xc4Q\xa7L\xb1\xe2Q\xf7h\xea\x10\xb7d\xf9(\x14;\xa0U\x1de\xf9AvX\xa9\xcbXt\rm\n\x99\x9b\xd6G\xcb\xf7\xe5\xc3\xe7ID\xaf\xa8Y<\xda\xc0'
```

How to import this TGT blob into the cred cache?

- Suggestions?

Next Steps

What’s next for this project?

What’s next for this project?

- Windows Hello with bio-metrics
  - Finger print reader
  - Probably won’t do facial recognition
- GDM Browser sign-on
  - Alexander has spoken about this previously
- Intune enrollment and policy application
  - Use augeas for policy enforcement
- Winbind integration
  - Again, help is needed to get WAF working with Rust!
- Proper idmapping
- Browser single-sign-on
  - xdg-credentials-portal (Webauthn platform API)

Demo

Demo using the Rust MSAL library

Thank You

For more information, contact SUSE at:

+1 800 796 3700 (U.S./Canada)

Maxfeldstrasse 5

90409 Nuremberg

www.suse.com