Microsoft

# The compromise of credentials

Threats targeting the hybrid & cloud identity platforms

# External resources disclaimer

This material includes links to external publicly available articles, projects, and research papers which are provided to you as a convenience and for informational purposes only.

Microsoft bears no responsibility for the accuracy, legality, content or any other aspect of the external site. Use of external hyperlinks does not constitute an endorsement by Microsoft of the linked content.

The external content referenced in this document belongs exclusively to their respective author(s). Inclusion in this presentation does not grant you with any right on the external content. You must comply with the original source's applicable policies.

# How to use this document

**Why this document?**

This document is provided as a companion of the video lessons. Additional information is included here which would not fit the video format or would exaggeratedly lengthen the videos. As you are watching the videos, the instructor will point you to additional content in this document.

**Structure**

The structure of this slide deck follows the structure of the lessons. One slide deck is provided for each module. The slide deck has the same structure (naming of chapters and sections) as the associated video so that you can quickly jump to the slides of the lesson you are currently watching.

# Foreword

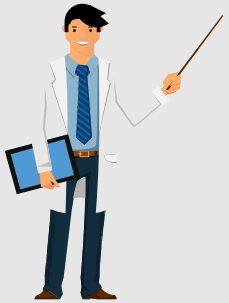
## This deck contains some design artefacts which all have their importance…


This sticky note icon is used to introduce the **abbreviation** of a concept or a technical word. Once the abbreviation has been introduced, the full version is no longer mentioned.

You will also find a list of all abbreviations at the end of the deck.

We were all young once.
A section with this icon will tell you the **history** you might have missed by not working with the technology for the last 20 years.

Just because you are new does not mean you do not have to know how we got here!



Professor Useful will introduce some **tricky technical details** which might not seem relevant at first but could end up being really useful if you want to dig deeper in the technology.

### This frame contains…

- Takeaways so important that we framed them

# How to know the slide level

This deck contains 3 different content levels:

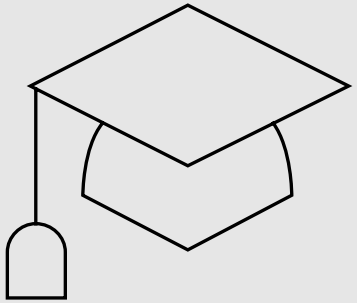
1. Regular level, the common slide
2. Advanced level, a slide with this indicator at the top left Adv.
3. Additional content, all hidden slides

Sequence

# 3

## The compromise of credentials

# Learning Objectives



Protecting an environment against credential theft.

# Agenda

1. Brute force password attacks
2. Password spray attacks on passwords
3. Kerberos roasting attacks
4. Abuse of user consent in Azure AD
5. Phishing attack with Device Code

Chapter

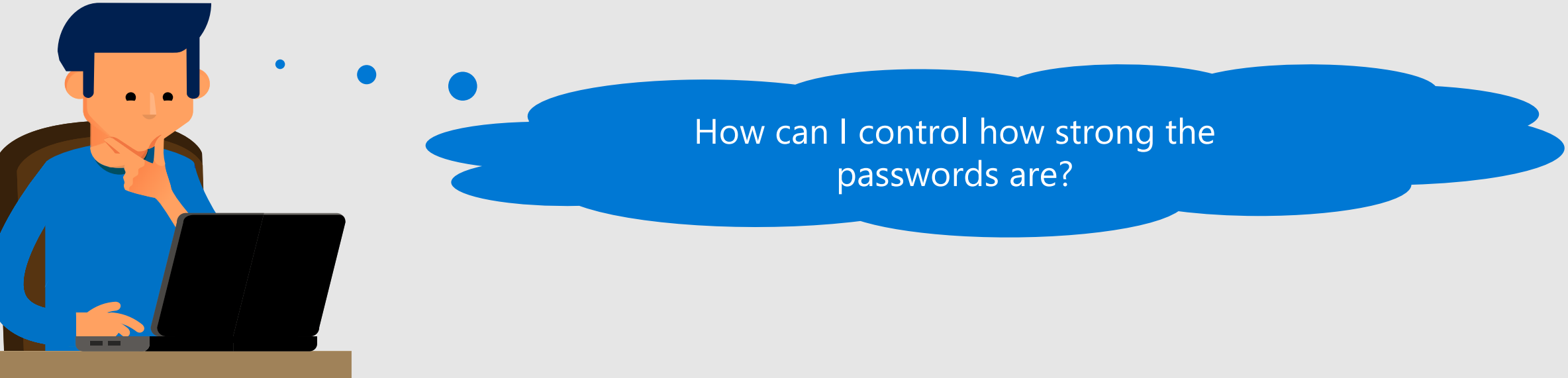
# 2.3.1

## Brute force password attacks

🎯 Develop a plan to secure the environment against brute force attacks.

# Brute force attacks

- In a brute-force attack, an attacker tries many passwords for one or many users
- It’s slow, it’s noisy, in fact it’s rarely used by attackers
- The weaker the password, the easier it is for the attacker



Adv.

# Brute force usernames with Kerberos

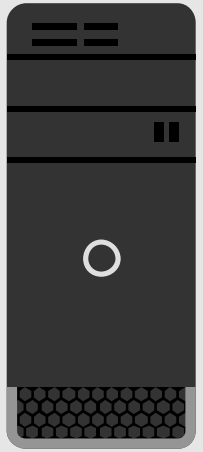
❌ bob
✅ alice

KRB_AS_REQ I want a ticket, I am **bob@contoso.com**

KRB_AS_REP Security principal unknown

KRB_AS_REQ I want a ticket, I am **alice@contoso.com**

KRB_AS_REP Pre-authentication required



Domain Controller

Adv.

# Brute force usernames with NTLM

❌ bob
✅ alice

I want to access the data →

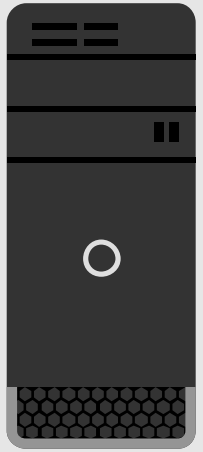
← Here is a challenge

Here is username **bob** and my response →

← NoSuchUser (0xc0000064)

Here is username **alice** and my response →

← WrongPassword (0xc000006a)

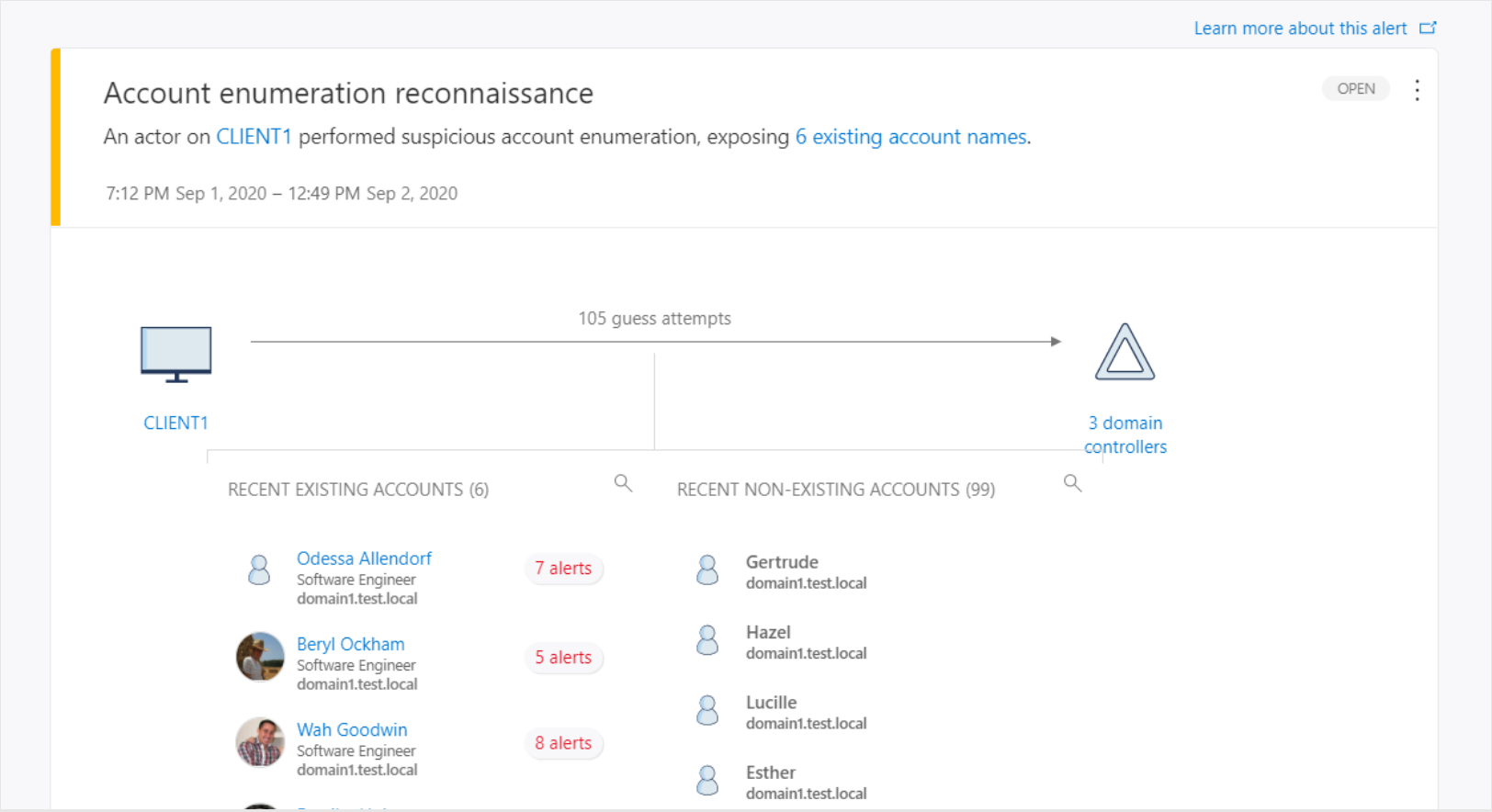


Server

Adv.

# Detection of enumeration on domain controllers

- Example of alerts from Microsoft Defender for Identity



# Brute force valid usernames using Kerberos

## Using NMAP

```
nmap.exe -p 88 --script krb5-enum.users --script-args krb5-enum-users.realm="
contoso.com"
```

# Password Policies

- Defines the password requirements

  - Password minimal length
  - Password minimal age
  - Password maximal age
  - Password complexity
  - Password history

FGPP

- There is a default policy that applies to everyone, and you can create multiple Fine Grained Password Policies and target identities

# Account Lockout Policies

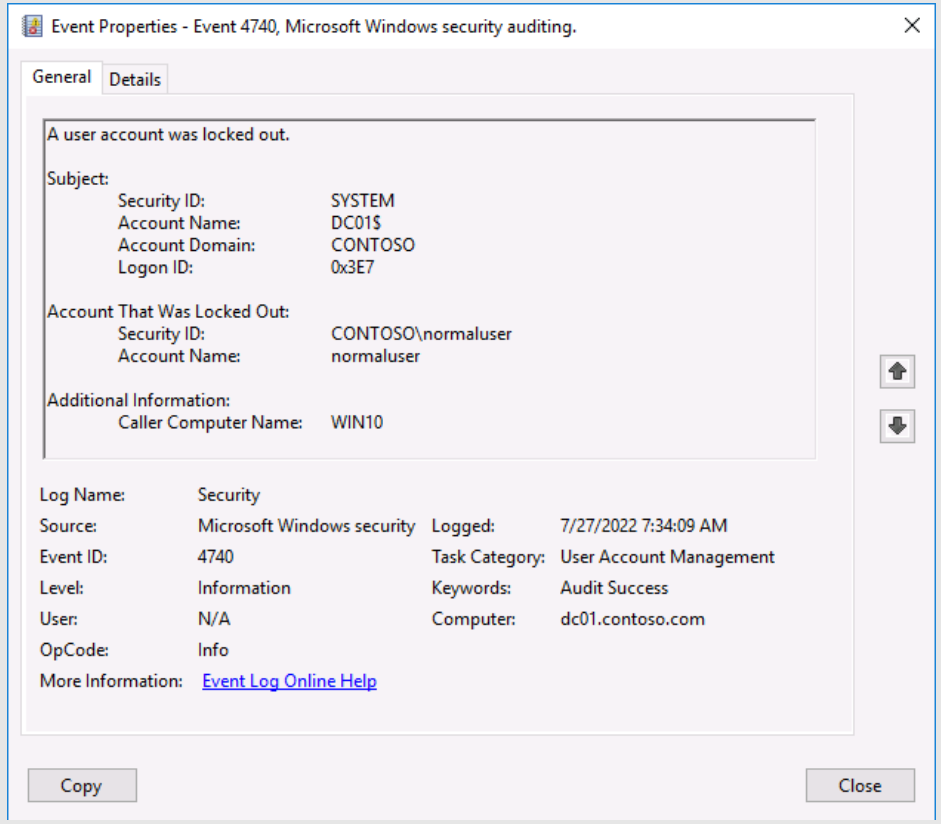
- Defines how many attempts before we lock the account and for how long

  - Account lockout threshold
  - Account lockout duration
  - Reset account lockout counter after

- You can also have multiple policies with FGPP

# Account Lockout Monitoring

- Events are generated on domain controllers when accounts are locked (event ID 4740)



Adv.

# Account Lockout Monitoring

- The **Caller Computer Name** field is not reliable
- It can be spoofed by an attacker misleading administrators in their investigations

- Examples of misleading situations:
  - In NTLM authentication, the field is showing the UserWorkstation attribute of the transaction which is provided at the discretion of the client (as attacker can spoof it)
  - In LDAP simple binds, the field will contain the name of the domain controller

# Account Lockout Policies

⚠️ An attacker can lock out all the accounts

- Plan for this eventuality
- The built-in administrator account can still be used even if it is locked out

# Password Policies back in the day

- Until Windows Server 2008

  1 domain = 1 password policy (before FGPP)

- If you want a different password policy for some users, you need a separate domain

- Lead to creation of empty root domains for enterprise admins

# Complex passwords might be a bad idea…

- **Passwords have a cost** 💰
- They are forgotten and need to be reset
- They are mistyped, locking out accounts
- Once a user found a good complex one, it is reused on multiple platforms
- They might be written on notes (unencrypted files or even on paper)

What can we do in the meantime we get rid of them?

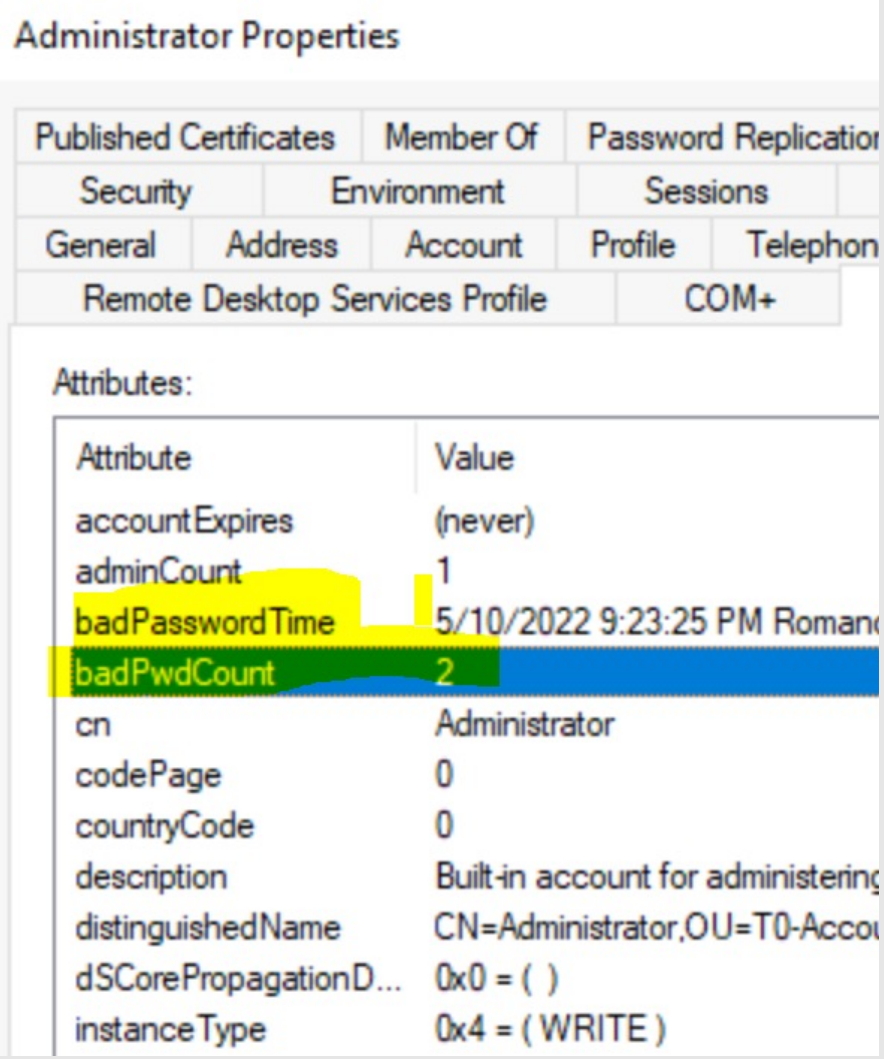
# What does research show?


1. Maintain an 8-character minimum length
2. Do not require complexity
3. Eliminate mandatory periodic password resets for regular user accounts.
4. Ban common passwords, to keep the most vulnerable passwords out of your system
5. Educate your users not to re-use passwords
6. Enforce registration for multi-factor authentication
7. Enable risk based multi-factor authentication challenges

Adv.

# Bad passwords and AD DS attributes

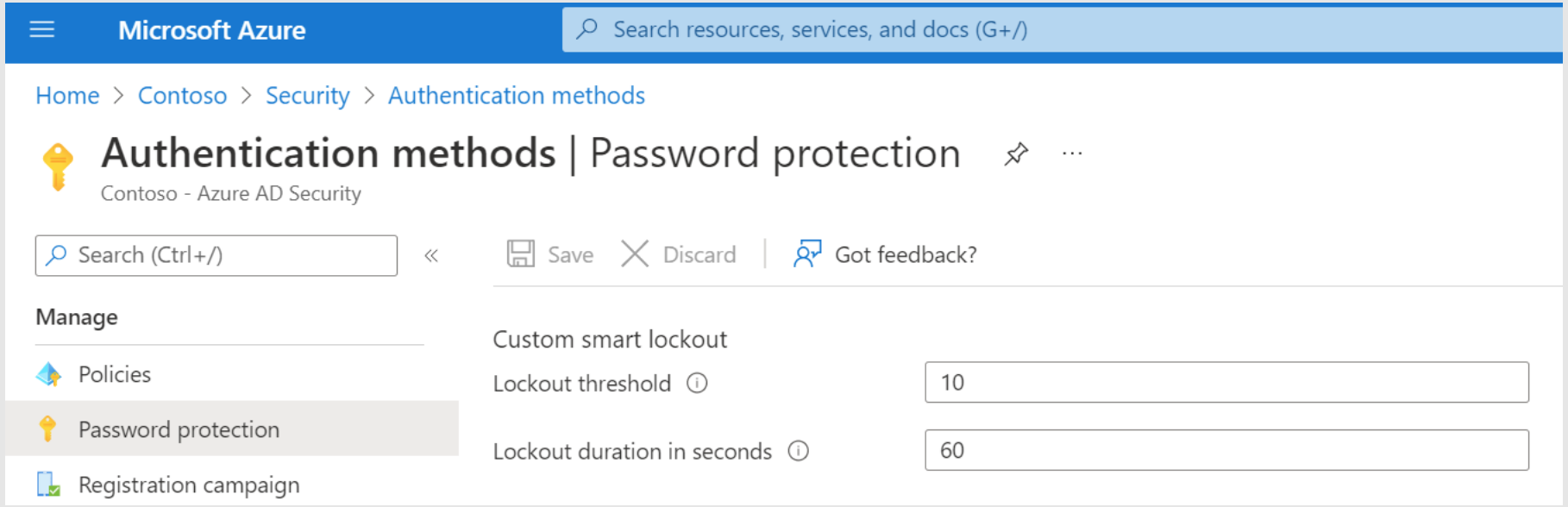
- When a bad password is used, the badPwdCount attribute is increased
- It indicates the number of bad password attempts
- The badPasswordTime indicates the last bad password attempt



# What about Azure AD?

- When the domain is managed, Azure AD has its own protection

**Smart Lockout**



Chapter

# 2.3.2

## Password spray attacks on passwords

🎯Protect a hybrid environment from password spray attacks.

# Password Spray, an intelligent Brute force?

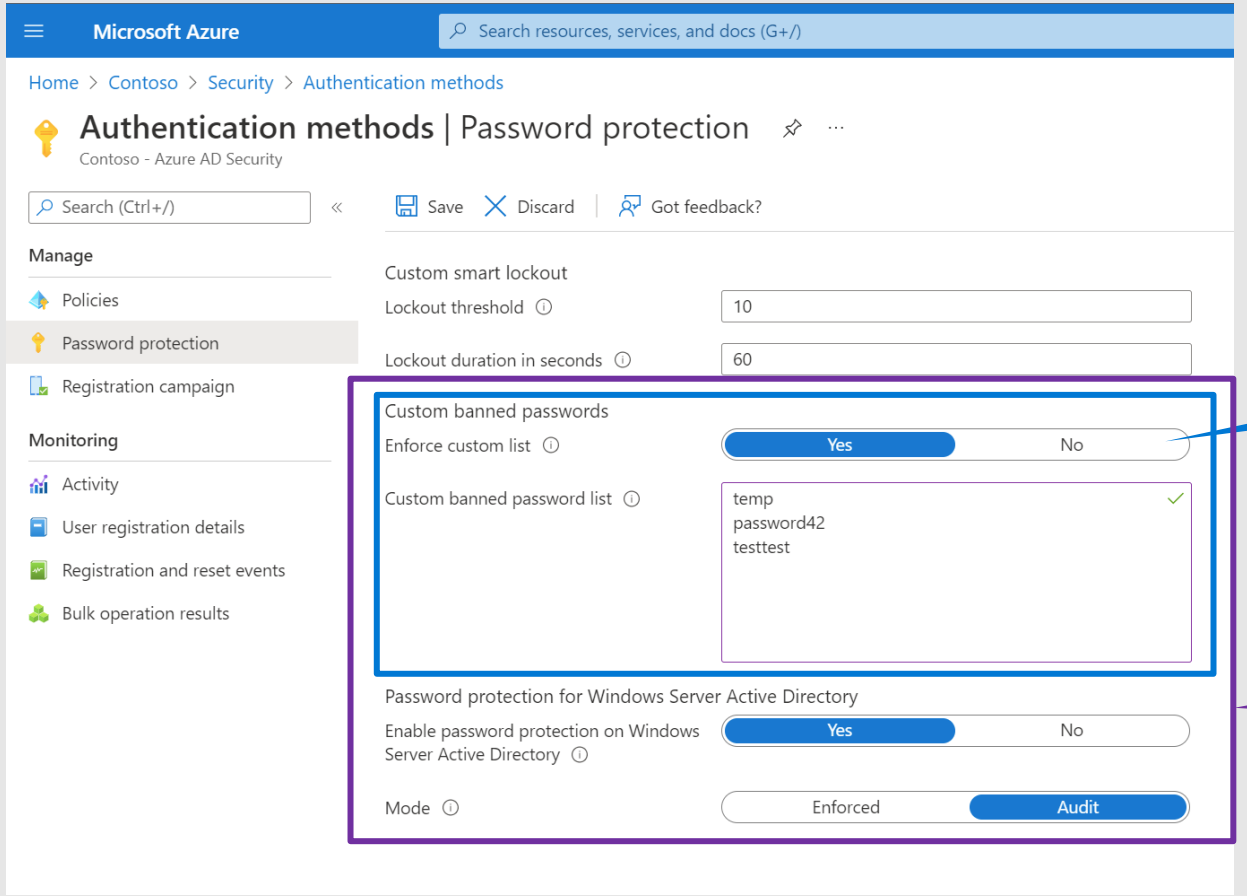
- Password Sprays are similar to Brute Force Attacks but more intelligent
- Only try a few common passwords for one or many users with the hope that at least one user is using one
- Could also try already compromised password lists found online
- Fly under the radar (doesn’t lock out accounts – usually)
- Examples:

  Password1

  Qwertyuiop!

  Hospital2022

# Then avoid using common passwords!




For cloud accounts

For synchronized identities
Applies on-premises on Domain Controllers

# Azure AD Password Protection

- Before, complexity is ON or OFF
  - With 3 out of 4-character sets
- Now, you can ban passwords that are easy to guess even if they are “complex”
  - It is a feature of Azure AD which can be back-ported to your on-prem AD
  - You can also customize a list of passwords you want to ban

# Password validation algorithm

- Global and custom lists are combined

All inputs normalized

All characters lower-cased

- Common character substitutions

‘$’ -> ‘s’, ‘@’ -> ‘a’, ‘0’ -> ‘o’, ‘!’ -> ‘1’, etc

- Fuzzy substring search (within edit distance of 1)
- Final scoring:

+1 for each banned token found

+1 for characters not part of banned tokens.

**Min score of 5 required to pass**

# Password validation example - failure

- User tries: “P@s$w0rD!2”

Banned passwords: “password”, “admin”

- Normalized to “password12” then:

“password” is found -> +1

‘1’ is found -> +1

‘2’ is found -> +1

❌ Total score: 3 (rejected)

# Password validation example - success

- User tries to change to “Admin!P@s$w0rd!3”

Banned passwords: “password”, “admin”

- Normalized to “admin1password13”

“admin” is found -> +1

“1” is found -> +1

“password” is found -> +1

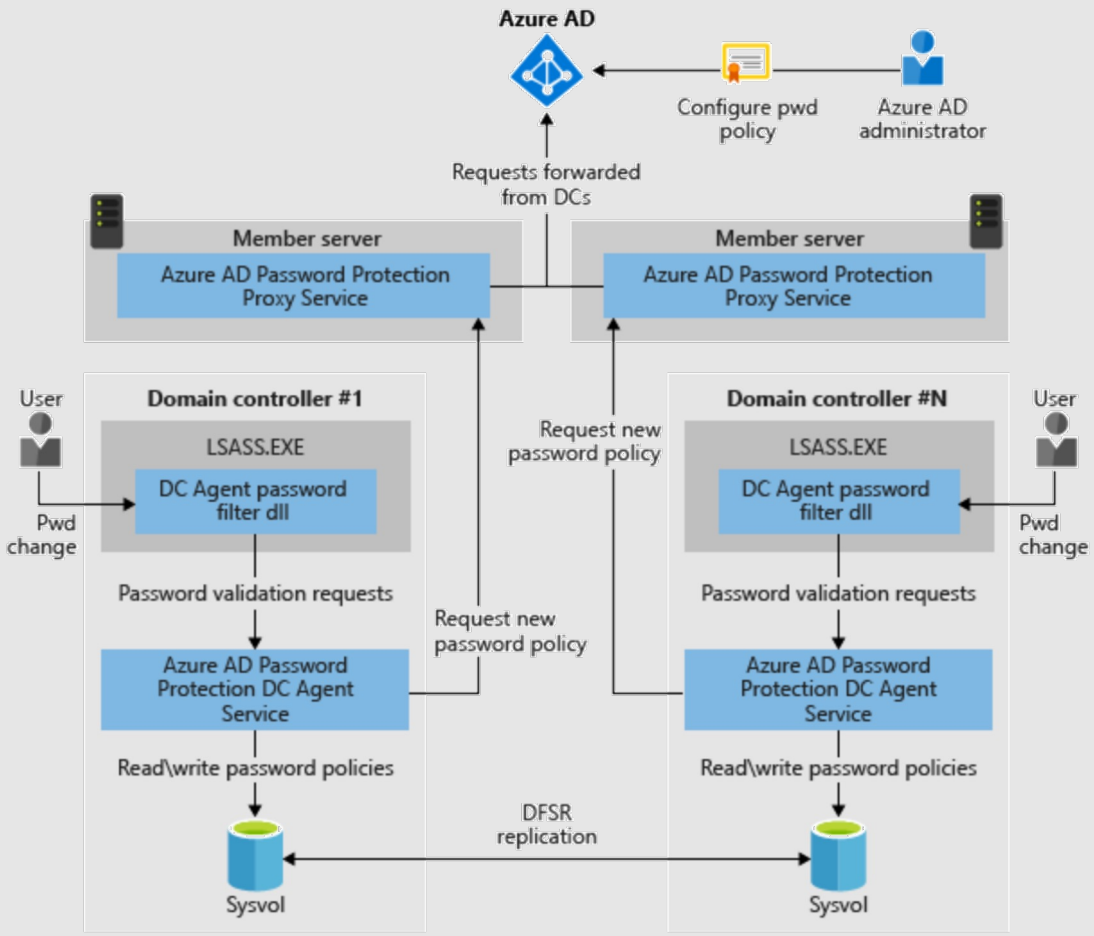
“1” is found -> +1

“3” is found -> +1

✅ Total score: 5 (accepted)

# Azure AD password protection for AD DS

- Requirements
  - Azure AD P1 licenses for all synced Users in AAD
  - Windows Server 2012 or higher
- Password policies stored in SYSVOL
- No need of internet connectivity for DCs
  - Only the proxy Agent needs one



Adv.

# Statistics

- Get-AzureADPasswordProtectionSummaryReport

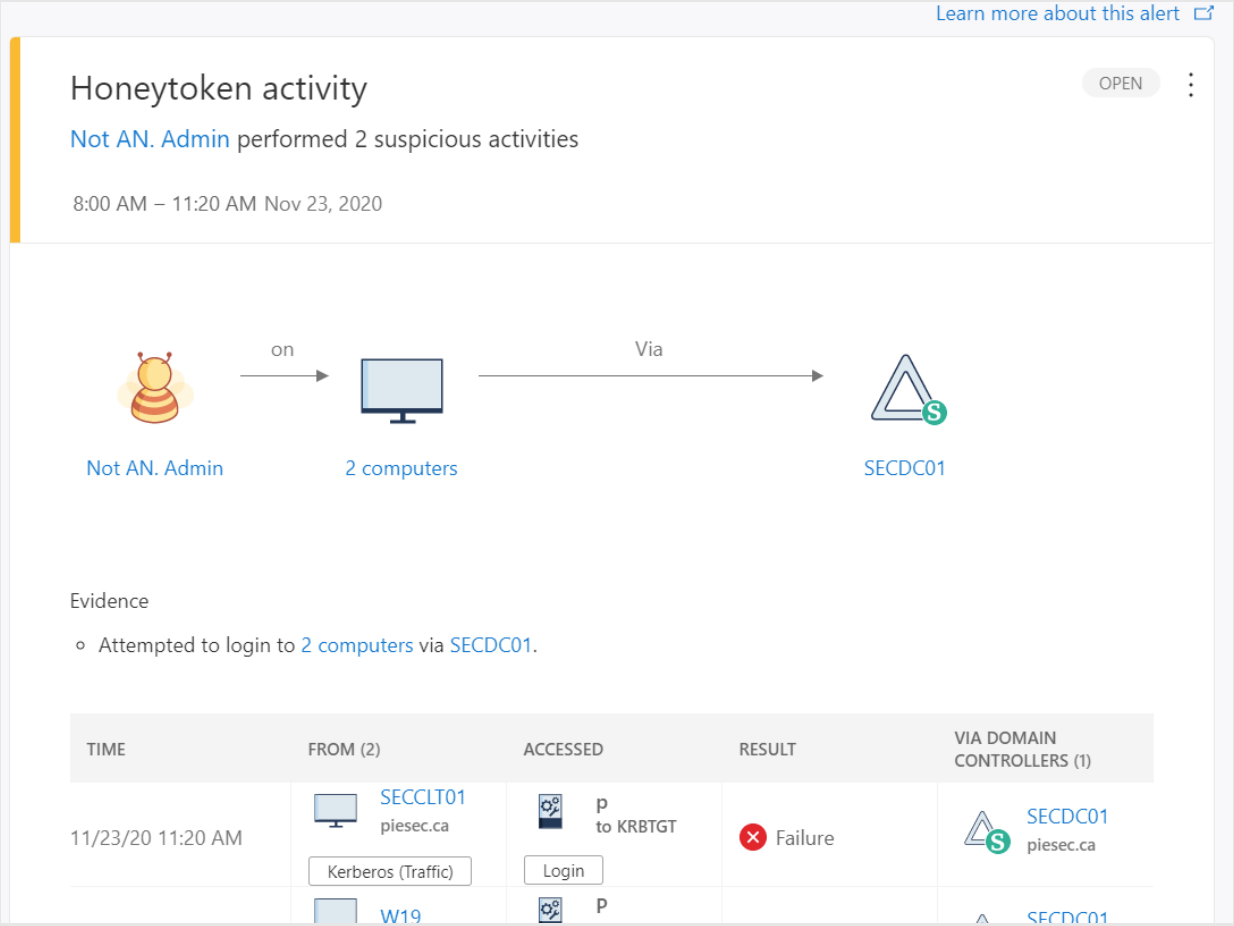
```
Get-AzureADPasswordProtectionSummaryReport -DomainController bplrootdc2
DomainController                : bplrootdc2
PasswordChangesValidated        : 6677
PasswordSetsValidated           : 9
PasswordChangesRejected         : 10868
PasswordSetsRejected            : 34
PasswordChangeAuditOnlyFailures : 213
PasswordSetAuditOnlyFailures    : 3
PasswordChangeErrors            : 0
PasswordSetErrors               : 1
```

# Honey Token Accounts

- Fake accounts which look like real ones
- Any attempts to use those accounts are suspicious
- Monitor logon attempts on domain controllers and alert
- Act on the source IP/Computer

# Honey Token Account alerts

- Example of alerts from Microsoft Defender for Identity



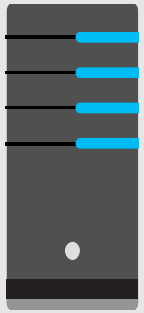
# Why is it sometimes so hard to identify the source?

Attacker IP: 1.2.3.4
Username: bob

NTLM Authentication
spoofed source = FUBAR

Failed

Open RDP
Server
Name = Server1

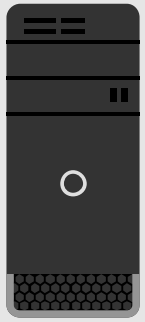


**Event 4625**
User = Bob
IP = 1.2.3.4

NTLM Passthrough

Failed

Domain
Controller



**Event 4776**
User = Bob
Source = FUBAR

⚠️ Not enabled by default
Use this one to identify the server

**Event 8004**
Server = Server1

Chapter

# 2.3.3

## Kerberos roasting attacks

🎯 List actions to protect AD against Kerberos roasting attacks.

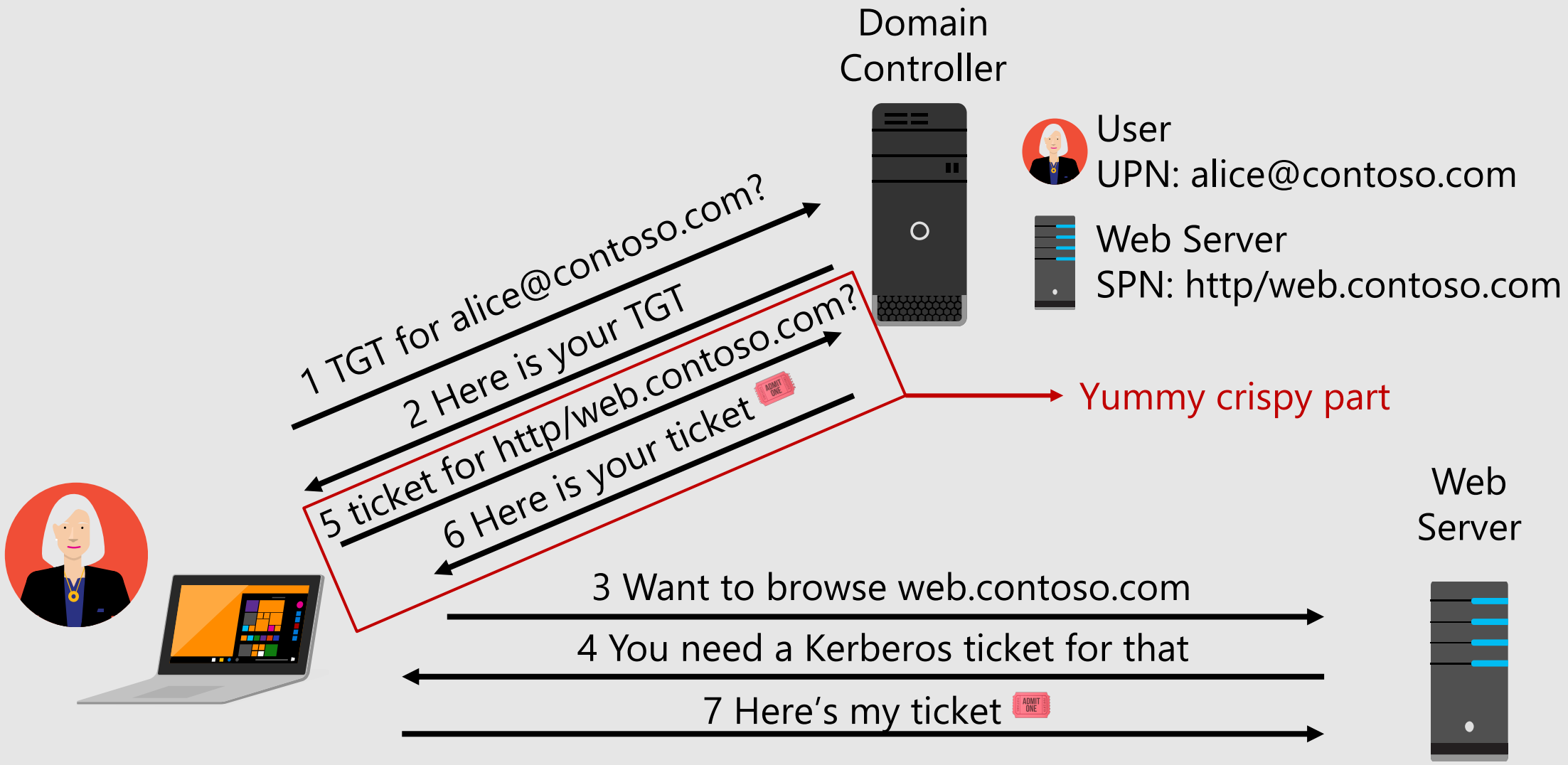
# How to brute force without suspicion?

- It doesn’t generate failed logon attempts (flies under the radar)
- Attacks the Kerberos tickets, not the accounts!
- You need a normal account

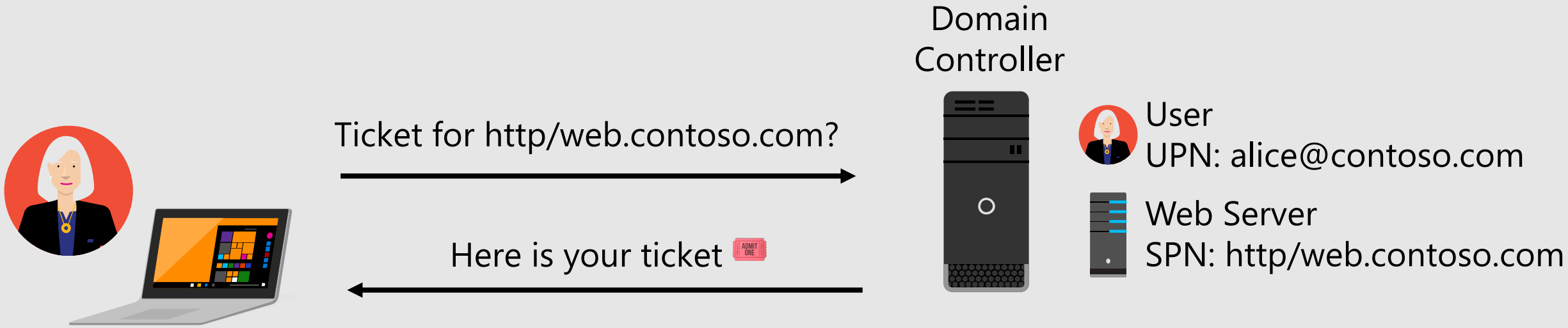
ℹ️ It’s an on-premises thing

ℹ️ You can use your own infrastructure to crack it offline

# How does Kerberos work?



# Let's zoom in




Is encrypted with the hash of the account on which the SPN is set
Is using either 3DES, RC4-HMAC or AES256 for the encryption

- An attacker will often request the lowest encryption type
- An attacker will try to crack the ticket offline

# Kerberos roasting attack MO

Steps to gain access by using Kerberos protocol

1. Enumerate accounts with SPN using LDAP (ideally account configured to support only RC4 or lower)
2. Request a ticket (requires a regular authenticated account)
3. Crack the ticket offline
4. Craft a custom ticket with arbitrary Privilege Attribute Certificate (with arbitrary group membership) aka **silver ticket**
5. Access the target service with the silver ticket

# Kerberos roasting attack detection challenges

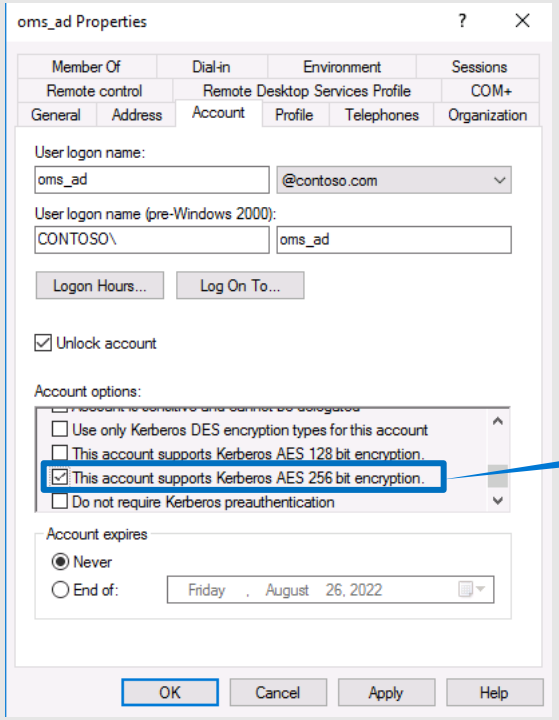
- The cracking takes place offline
- Does not generate any traces other than the original ticket request

But there are opportunities of detection!

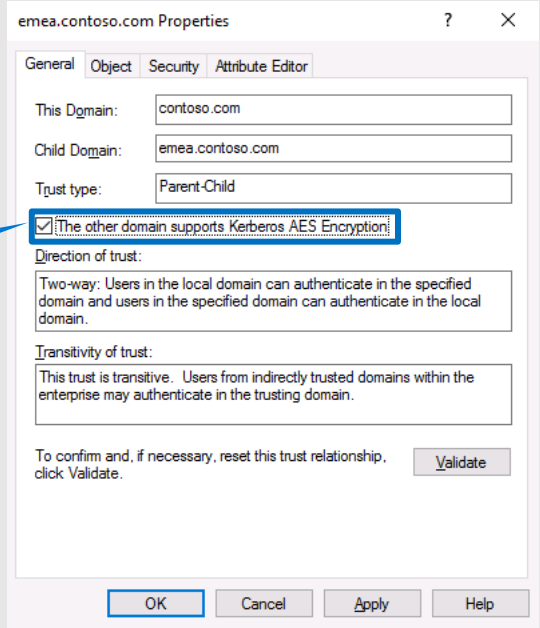
- Catch the LDAP enumeration of the attacker
- Catch the encryption downgrade ticket request

# Protecting the environment against Kerberos Roasting

- Ensure the conditions for AES256 are met
  - Requires actions on service accounts
  - Requires actions on domain trusts



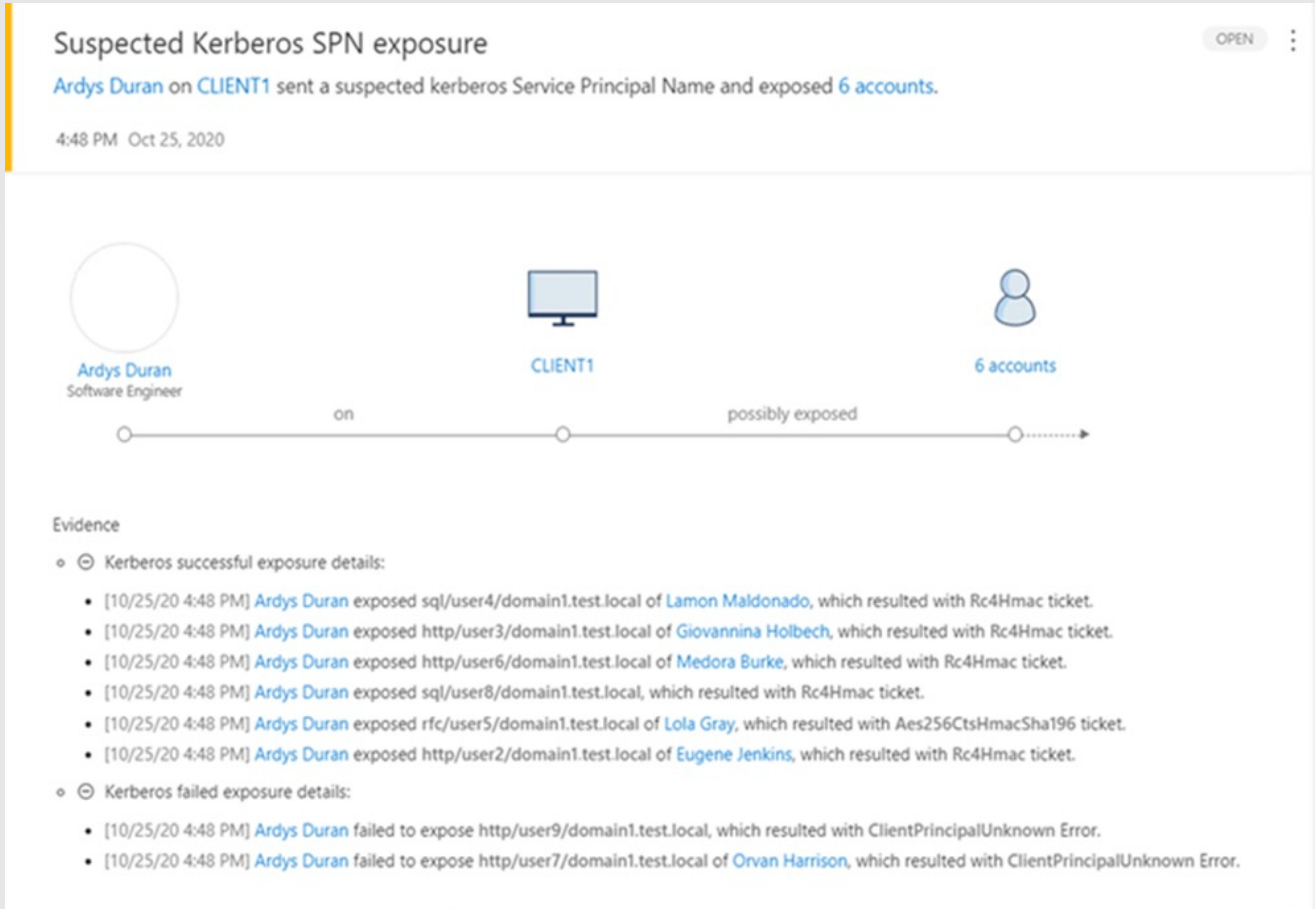

Not enabled by default




- Use long and complex passwords for service accounts or gMSA

# Detect SPN enumeration

- Example of alerts from Microsoft Defender for Identity

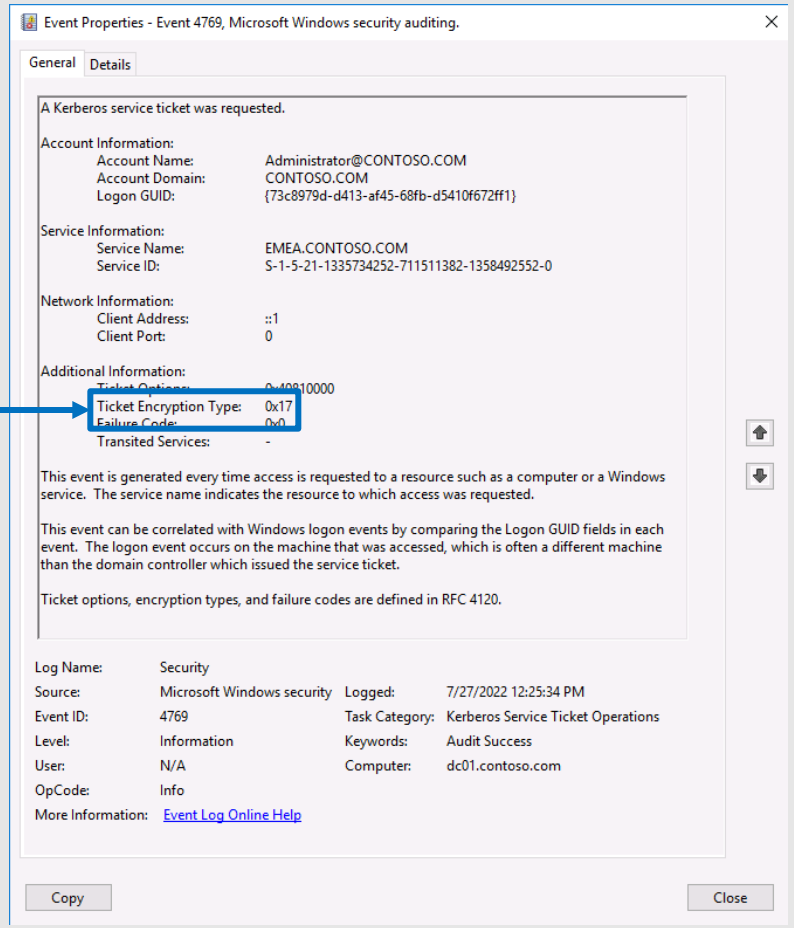


Adv.

# Detect ticket requests with weak encryption types

- In the security event logs of the domain controllers
- Event ID 4769

| Type | Type Name |
|---|---|
| 0x12 | AES256-CTS-HMAC-SHA1-96 |
| 0x17 | RC4-HMAC |



Adv.

# msDS-SupportedEncrytionType

- Attributes that govern what encryption types are used for tickets
- Updated by the machines themselves on computer accounts
- Needs to be set manually on user accounts (if they are service accounts, meaning they have a Service Principal Name defined)

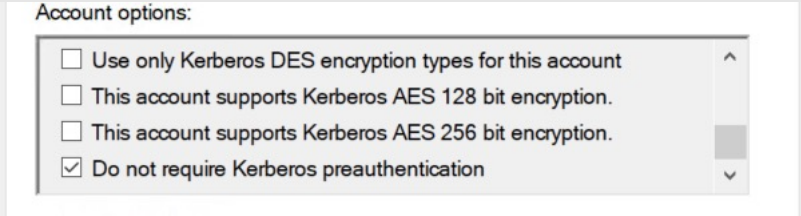
⚠️ Not all OS supports the highest encryption type

⚠️ Forcing AES256 may break applications

Adv.

# Roasting accounts without SPN

- Roasting can also work on principals without a service principal name attribute if the DONT_REQ_PREAUTH is on.




- In that case we can ask for a TGT and try to roast the encrypted results the same way an attacker would roast a service ticket

Adv.

# Enumerate accounts with DONT_REQ_PREAUTH

## LDAP filter

```
(&(objectCategory=person)(objectClass=user)(userAccountControl:1.2.840.113556.
1.4.803:=4194304))
```

# Group Managed Service Account

- **gMSA** for short GMSA
- The password is managed by the domain controllers
- It’s long, complex, random, and changes every 30 days
- Very unlikely to be found in a Kerberos Roasting attack

✅ No need to create an account with a password that never expires
✅ Can’t be used interactively, so users can’t use the account
⚠️ The application needs to be compatible

Chapter

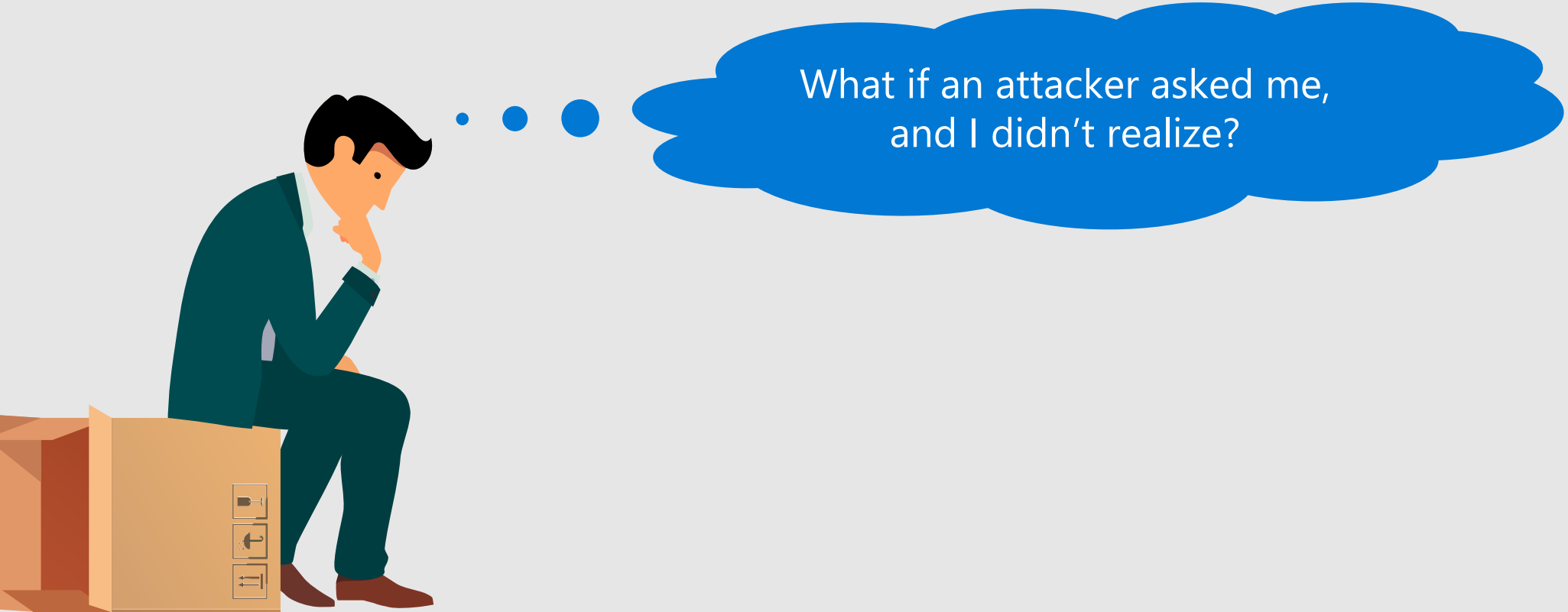
# 2.3.4

## Abuse of user consent in Azure AD

🎯Describe the OAuth2 mechanisms involved in consent

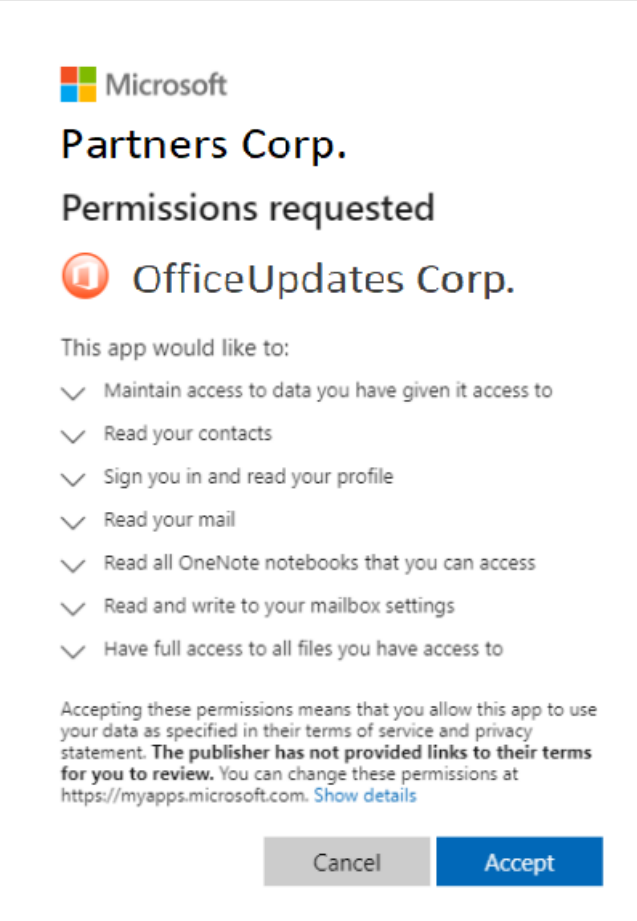
# Reminder

- **Consent** is the process of a user granting authorization to an application **to access protected resources** on their **behalf**.



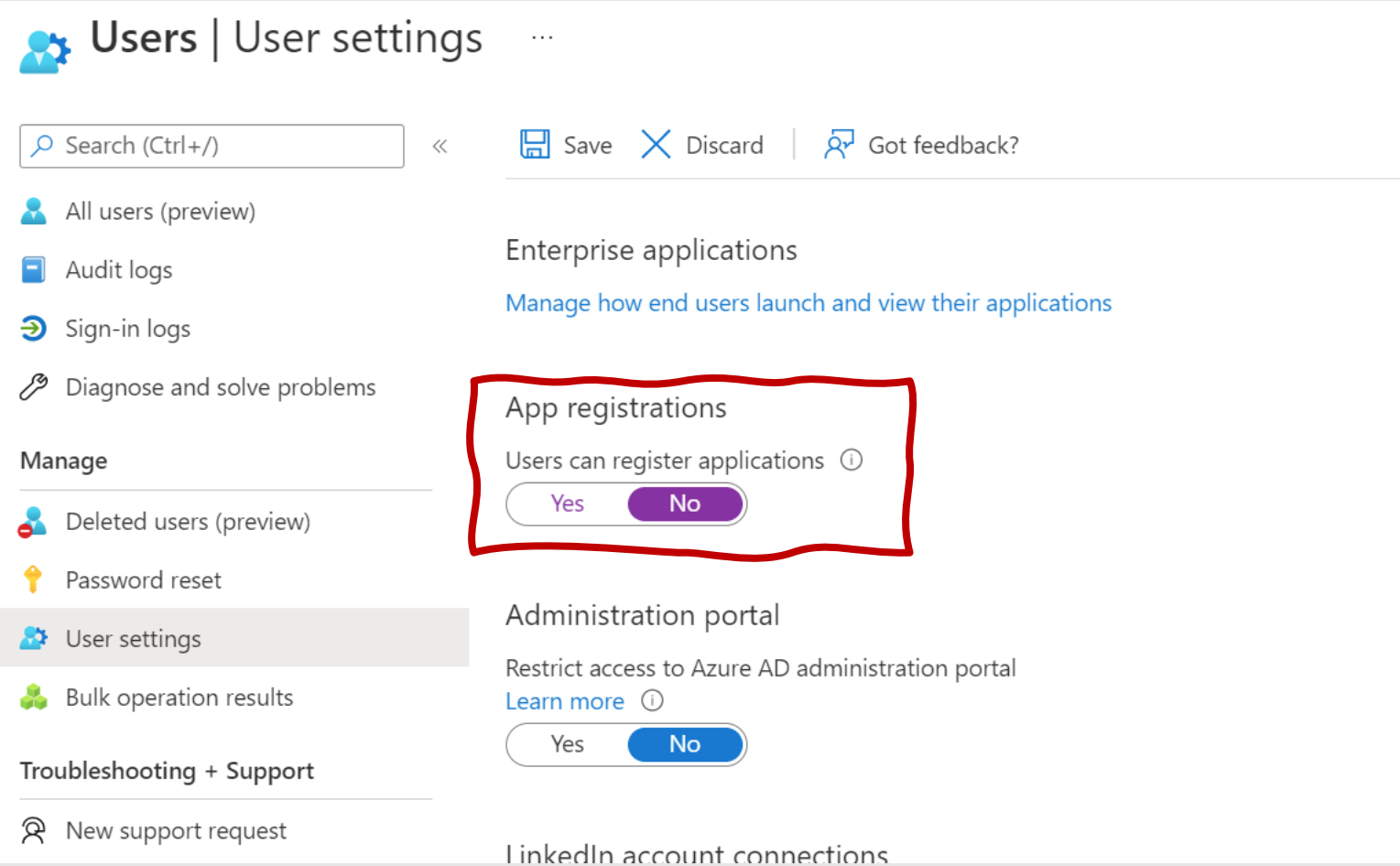
# Abusing consent MO

1. The attacker creates a multi-tenant application in its own tenant
2. Configures the application to be granted delegated access
3. The attacker tricks the user to connect to the application (phishing)
4. If the user accepts, the attacker can access the user's resources



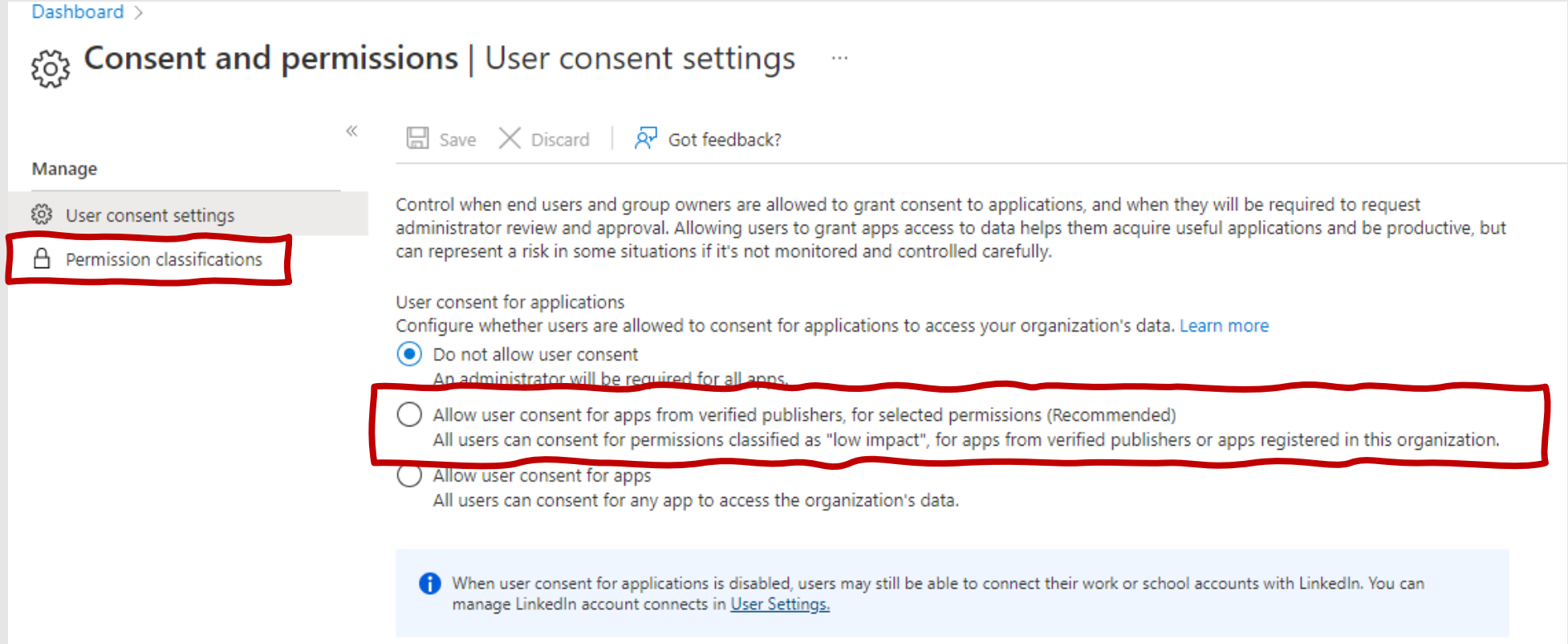
# Protection against illicit consent

- No consent from new application



# Protection against illicit consent

- Set up risk-based step-up consent and MPN Identifiers
- The admins can control the users' consent



Chapter

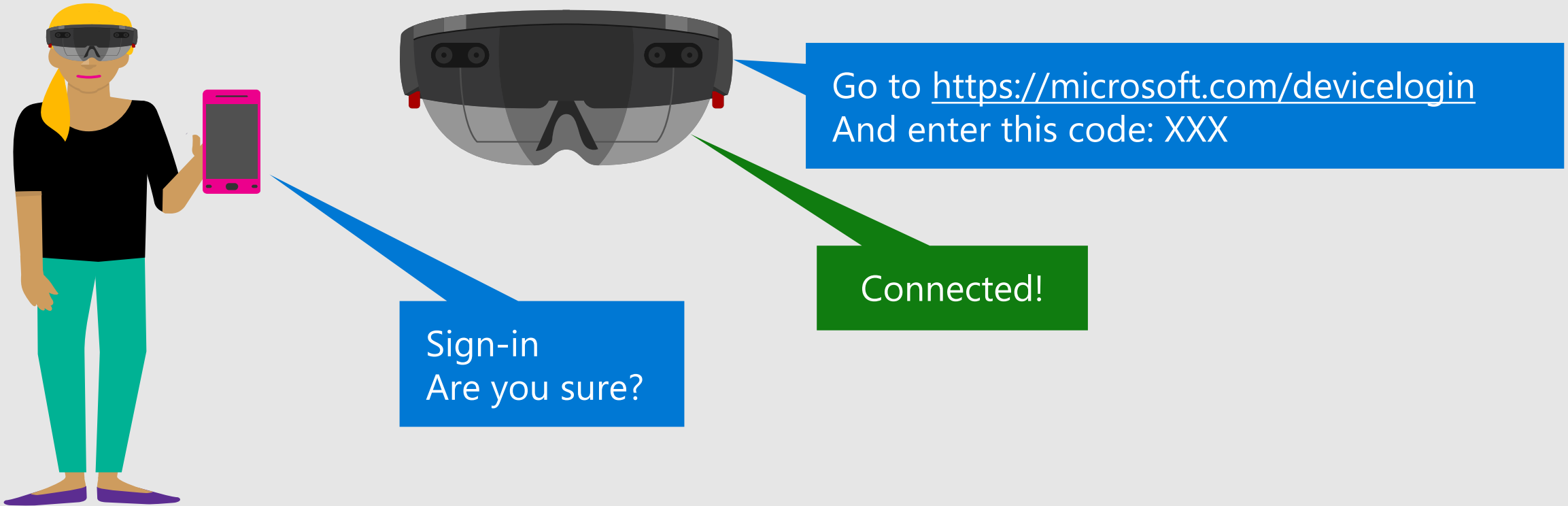
# 2.3.5

## Phishing attack with Device Code

🎯Describe the attack abusing Device Code flow in Azure AD

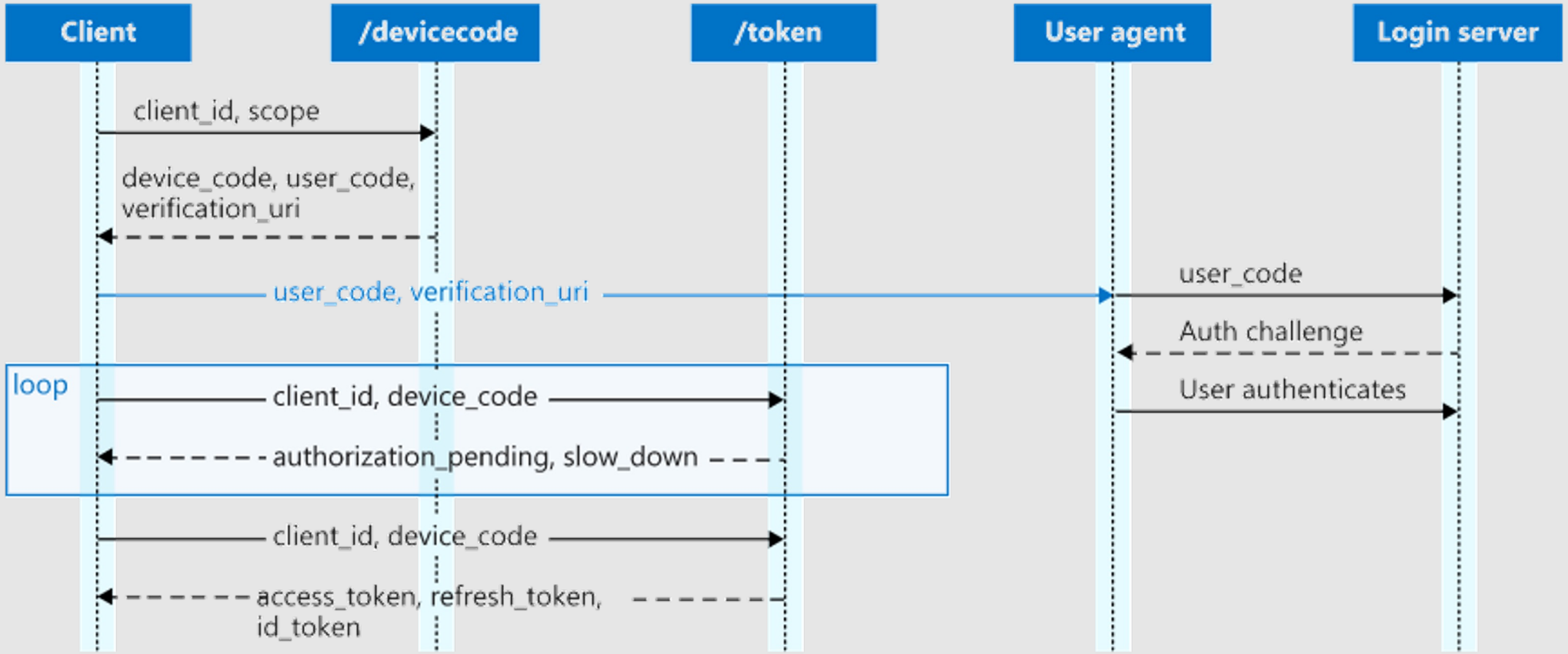
# Device code AuthN

- Authenticate users on input-constrained devices (such as IoT) or devices that don't provide a web browser



Adv.

# Device code flow



# Abusing the device code flow MO

1. Try to access data using code flow and get a code
2. Trick the user into entering the code in the devicelogon page (phishing)
3. Connect on behalf of the user
4. Pivot to other backend resources and access other resources

# PowerShell tool to make Code Flow

## AAD Internals module (excerpt)

```
$body=@{
        "client_id" = "d3590ed6-52b3-4102-aeff-aad2292ab01c"
        "resource" =  "https://graph.windows.net"
}
$authResponse = Invoke-RestMethod -UseBasicParsing -Method Post -Uri
"https://login.microsoftonline.com/common/oauth2/devicecode?api-version=1.0" -Body
$body
$user_code =    $authResponse.user_code
Send-MailMessage ... <phishing email>
$response = Invoke-RestMethod -UseBasicParsing -Method Post -Uri
"https://login.microsoftonline.com/Common/oauth2/token?api-version=1.0 " -Body $body

# Dump the tenant users to csv
Get-AADIntUsers -AccessToken $response.access_token | Export-Csv users.csv
```

Microsoft

# List of abbreviations

FGPP – Fine Grained Password Policy

gMSA – Group Managed Service Account

SP – Service Principal