Active Directory passwords: All you need to know

All Windows administrators need to know the essential concepts of Active Directory passwords: how passwords are stored in Active Directory, how password authentication works, and how to manage Active Directory passwords. A common task for admins is to reset users' passwords, which you can do with the GUI or PowerShell. However, in large networks, a self-service solution is required. Computer account passwords are another topic that administrators have to be familiar with.

Contents

1. Hashing and encryption

Before directly discussing how passwords are stored in AD, I want to spend some time discussing various password security techniques such as hashing, salt and pepper, encryption, etc. If you already know about them, you can skip directly to the next section.

Hashing is totally different from encryption or encoding. It is an irreversible deterministic operation that transforms an input value into a fixed-length output, called a hash digest (or simply a hash). When we take an input string (e.g., a password) and pass it to a hashing algorithm (e.g., SHA256), it gives us a hash digest as an output. The thing about hashing is that it is a one-way process, which means that if someone gets their hands on your hashed password, they cannot (theoretically) determine your original password from it.

Encryption, on the other hand, is a reversible two-way operation, which means the output of the encryption algorithm (known as ciphertext) can be decrypted back to obtain the original input value (e.g., the actual password).

Now, you might start thinking that a password, once hashed, is unbreakable. Well, not really. Hashes can be reverse-engineered. Threat actors tend to steal password hashes and then run different types of attacks (brute force, dictionary, or rainbow table) for password guessing. All passwords can be cracked when given enough time and computing power. The older algorithms (such as MD4 or MD5) are quite weak and thus are easier to crack. Therefore, these are no longer recommended for use.

Salt and pepper are commonly used to safeguard against such attacks. *Salt* is a plain-text value that is appended to the original password before the password is passed to a hash function. The salt is usually stored in a database alongside a password hash, and it is helpful in thwarting rainbow table attacks. Similarly, *pepper* is a secret value that is either appended or used as a key to sign the original password value, which helps slow down brute-force attacks. Unlike salt, it is not stored in the database.

Modern hashing algorithms, such as SHA-256 or bcrypt, offer stronger protection against password-guessing attacks. Modern security standards suggest using a slower algorithm (like bcrypt) when it comes to password hashing rather than fast algorithms (like sha256 or sha512), since threat actors can't leverage modern computing resources (faster CPUs, GPUs, parallel processing, etc.) to speed up the password cracking process.

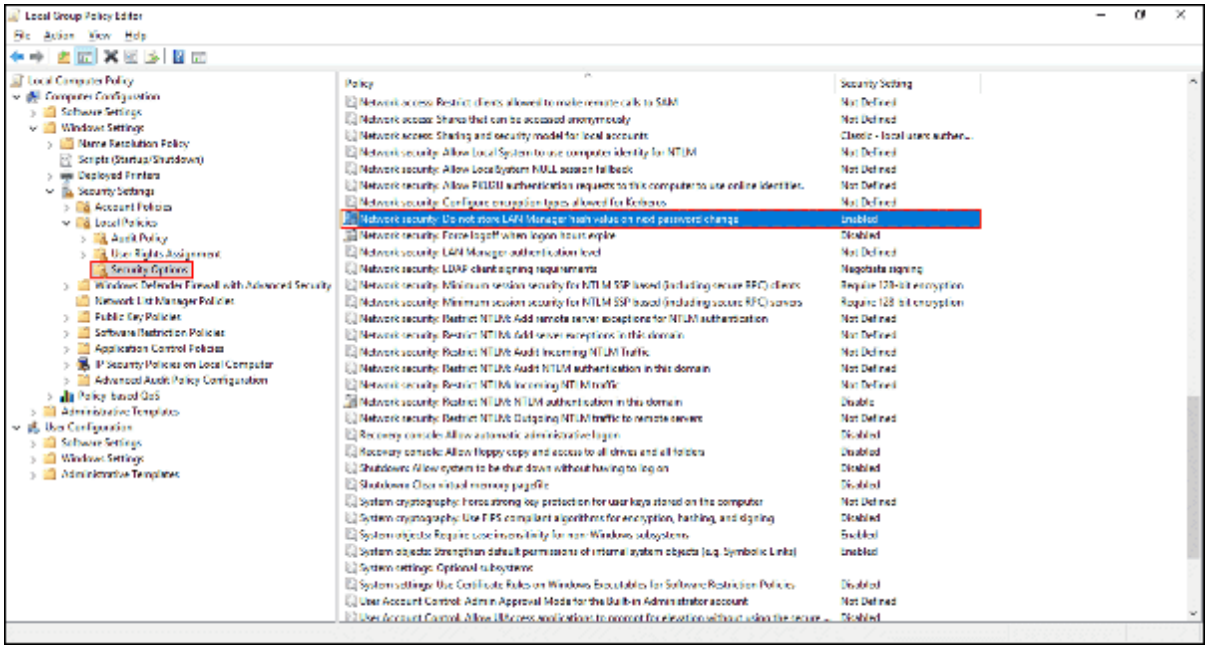
2. How are passwords stored in Windows?

Now that you understand the basic techniques, let's come back to the original question: how are passwords stored in AD? In both the Security Accounts Manager (SAM) database and the AD database (NTDS.DIT), passwords are stored as a hash digest. Microsoft calls the hashing algorithm a one-way function (OWF), but we will call it a *hash* for simplicity. Basically, when you change a user's password, it is stored in two different ways:

LAN Manager (LM) hash—The LM hash uses a really old hashing technique that supports a maximum password length of 14 characters (bytes), which is split in two halves of 7 bytes each. The LM hash was easier to break, so it has been

disabled by default, starting with Windows Vista and Windows Server 2008. It can be controlled by the *Network security: Do not store LAN Manager hash value on next password change* group policy setting, which is available at the following location:

Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options



Network security Do not store LAN Manager hash value on next password change

Network security: Do not store LAN Manager hash value on next password change

You can see in the screenshot that this setting is *enabled* by default to prevent storing the LM hash.

New Technology (NT) hash—The NT hash is a relatively newer technique that stores the MD4 hash of a password. We know that MD4 and MD5 aren't safe anymore, but that's what Microsoft uses for backward compatibility. Starting with Server 2016 and Windows 10, the NT hash is protected with two layers of security: it is first encrypted with DES and then with CNG bcrypt AES-256.

The interesting thing to note is that neither of these two hashes (LM hash or NT hash) is *salted* while stored in the AD database. The following table shows how passwords are stored in various attributes in AD:

AD attribute name	Stored as
unicodePwd	Encrypted NT hash
dbcsPwd	Encrypted LM hash
ntPwdHistory	Encrypted NT hashes
lmPwdHistory	Encrypted LM hashes
supplementalCredentials	Kerberos keys, Wdigest, Cleartext etc.

As I have already mentioned, LM hashes are disabled by default in newer Windows versions. Therefore, they are no longer stored in the AD database. The *supplementalCredentials* attribute, however, stores the password in clear text if the *Store passwords using reversible encryption* setting is enabled. However, this is a really bad idea, and this setting shouldn't be enabled.

3. How password authentication works in Windows

When a user types in their password, it is converted into two forms: LM hash and NT hash, as discussed above. They are then stored in memory by a protected Local Security Authority Subsystem Service (LSASS) process. If the user is a local account, then its NT hash is compared to that of locally stored hash in the Security Accounts Manager (SAM) database. If both hashes match, authentication is successful, and the user is logged in.

If a user is using an AD account, its NT hash is used in Kerberos authentication against a key distribution center (KDC)—a domain controller. Upon successful authentication, a Kerberos Ticket Granting Ticket (TGT) is issued and stored in the Kerberos ticket cache. The TGT can then be used to request additional access tokens from the Ticket Granting Service (TGS) for a particular network resource. You can use the built-in *klist* command to get information from the Kerberos ticket cache.

```
Administrator: C:\Windows\system32\cmd.exe
C:\Windows\system32>
C:\Windows\system32>klist tickets

Current LogonId is 0:0x57ba0

Cached Tickets: (2)

#0>     Client: Surender @ TESTLAB.LOCAL
        Server: krbtgt/TESTLAB.LOCAL @ TESTLAB.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
        Start Time: 10/17/2022 2:08:02 (local)
        End Time:   10/17/2022 12:08:02 (local)
        Renew Time: 10/24/2022 2:08:02 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called: SRV101

#1>     Client: Surender @ TESTLAB.LOCAL
        Server: LDAP/SRV101.TESTLAB.LOCAL/TESTLAB.LOCAL @ TESTLAB.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent ok_as_delegate name_canonicalize
        Start Time: 10/17/2022 2:08:05 (local)
        End Time:   10/17/2022 12:08:02 (local)
        Renew Time: 10/24/2022 2:08:02 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0
        Kdc Called: SRV101.TESTLAB.LOCAL

C:\Windows\system32>
```

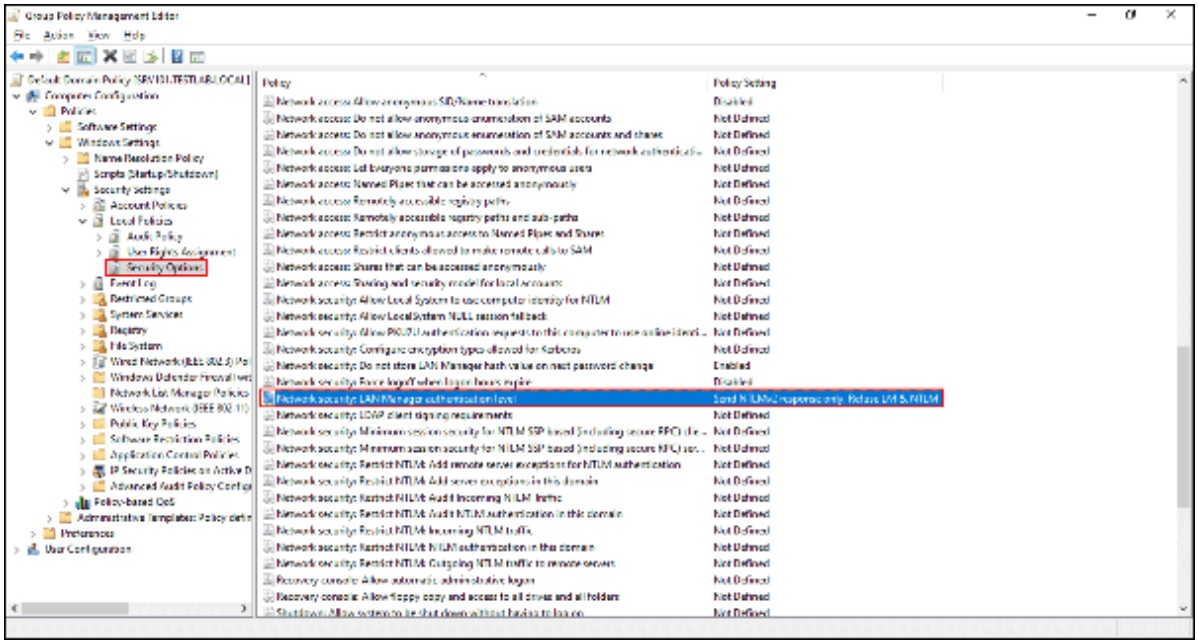
Use the klist command to view cached Kerberos tickets

The *klist tickets* command can be used to list all cached Kerberos tickets. However, there are certain scenarios in which Kerberos authentication cannot be used:

- Using an IP address to access a resource on an Active Directory domain member
- Accessing a resource on a non-domain-joined computer
- Accessing a resource on a computer that doesn't support Kerberos
- Authenticating against a Windows NT 4.0 or earlier domain

In such cases, LAN Manager (LM) and New Technology Lan Manager (NTLM) challenge–response protocols are used. The LAN Manager makes use of the old and vulnerable LM hash, whereas NTLM makes use of the NT hash.

Frankly, both the LM and NT hashes are vulnerable, so it is always recommended to use Kerberos authentication whenever possible. If you really have to fall back to NTLM authentication, however, always use the newer version (NTLMv2), as it offers better protection against relay and brute-force attacks. NTLMv1 is very weak, so you can disable it in your domain by setting the value of *Network security: LAN Manager authentication level* group policy to *Send NTLMv2 response only, Refuse LM & NTLM* as shown in the following screenshot:



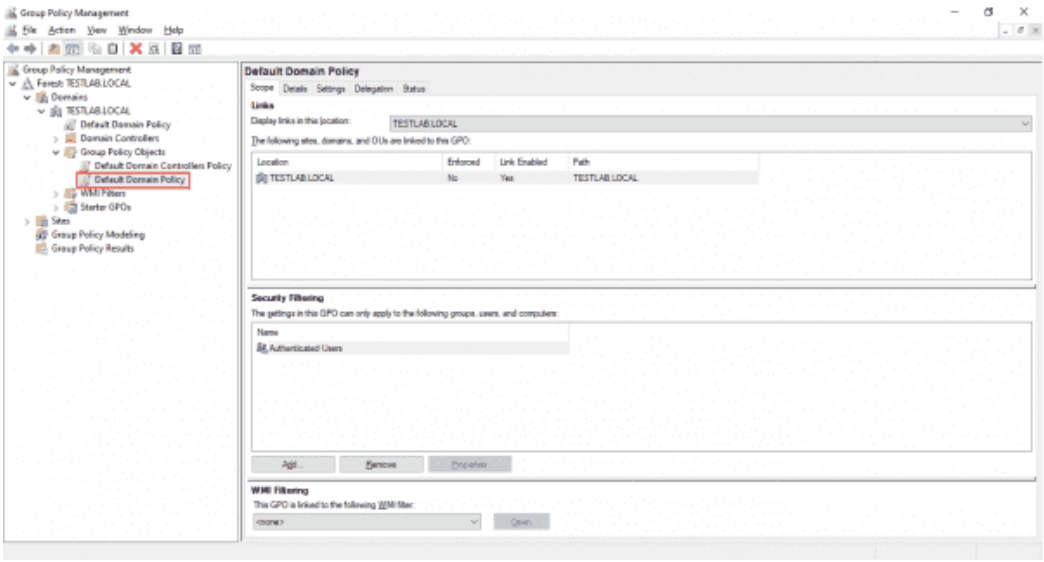
Use Network security LAN Manager authentication level group policy to disable LM and NTLMv1 in domain

Before directly enabling this policy domain-wide, I would recommend that you enable the auditing of NTLMv1 traffic in your domain, analyze the audit logs, find out which devices are still using NTLMv1, and then assess the overall impact of disabling NTLMv1.

4. Manage the Active Directory password policy

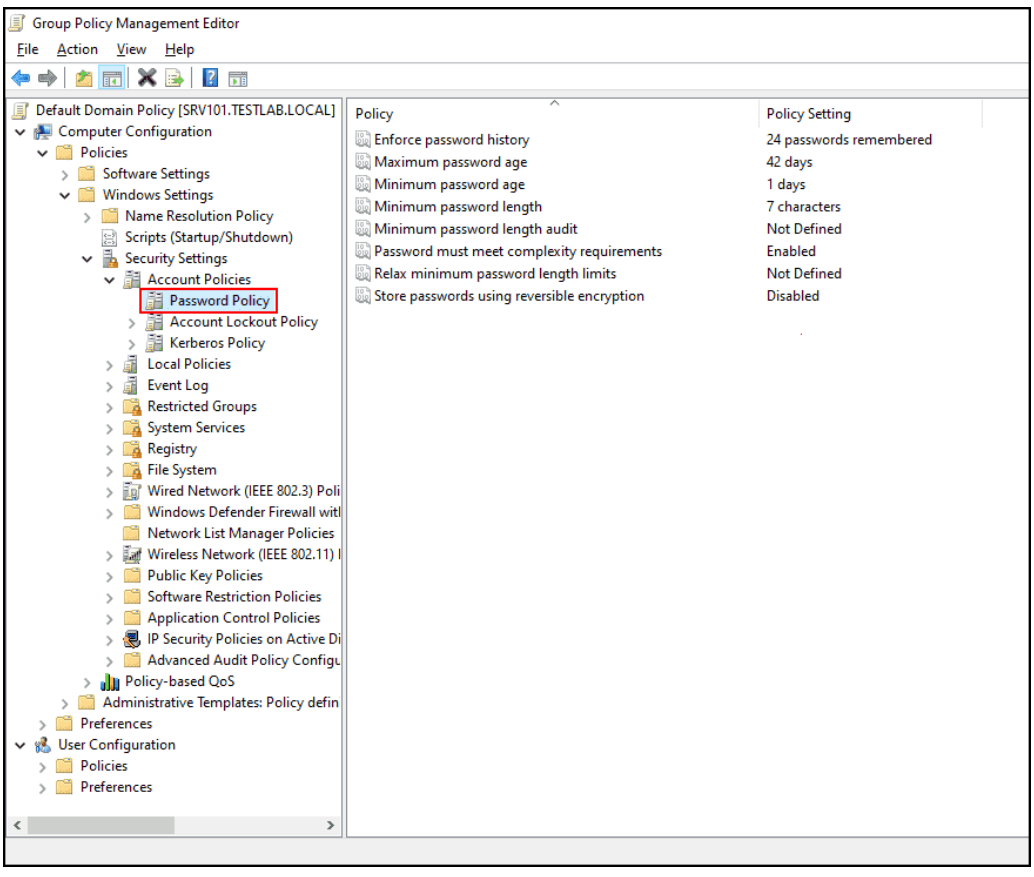
There is a default password policy in AD to control how passwords are managed throughout the domain environment. You can view or manage this policy by following these steps:

1. Launch the Group Policy Management console (gpmc.msc) on a domain controller.
2. Navigate to the *Default Domain Policy,* as shown in the screenshot:



Locating default domain policy in group policy management console

3. Now right-click the *Default Domain Policy* group policy object and select *Edit*. This will open the Group Policy Management Editor in a new window.
4. Now, navigate to the following path to view the password policy:*Computer Configuration\Policies\Windows Settings\Security Settings\Account Policies\Password Policy*



Locating the default domain password policy

Here, you can view and change the default password policy for the domain as per your organization's requirements.

5. Default password policy settings

Below are the settings for the default password policy in an AD domain running on Windows Server 2022:

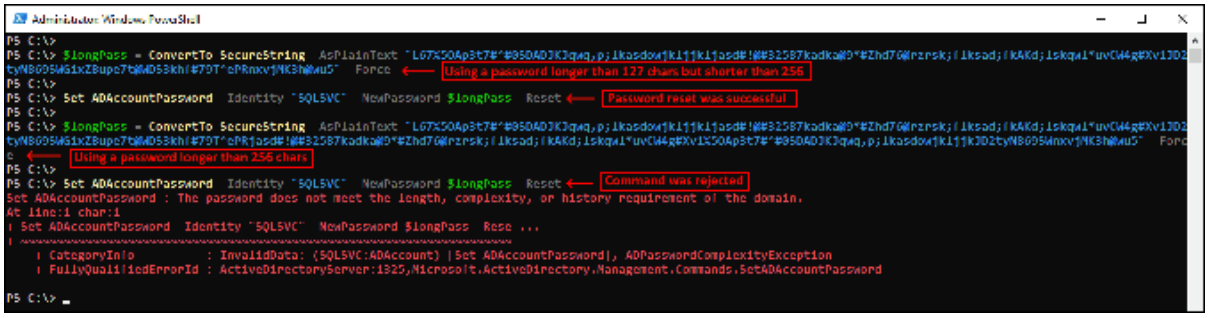
- **Enforce password history:** 24 passwords remembered
- **Maximum password age:** 42 days
- **Minimum password age:** 1 day
- **Minimum password length:** 7 characters
- **Minimum password length audit:** Not defined
- **Password must meet complexity:** Enabled
- **Relax minimum password length limits:** Not defined
- **Store passwords using reversible encryption:** Disabled

All of these settings are pretty much self-explanatory, and the default password policy is good enough for most organizations. However, admins can customize the default password policy or even use fine-grained password policies to meet organizational requirements when necessary.

6. Maximum password length in Active Directory

If you take a look at the default password policy settings, there isn't any option to control the maximum password length. So, what is the maximum supported password length in AD?

AD supports a maximum password length of 256 characters. Unfortunately, however, Windows GUI tools only allow you to type a password with a length of up to 127 characters. To set a longer password than this, you can use PowerShell or some programmatic method. Such long passwords are pretty uncommon but are still useful for service accounts. See the following screenshot for reference:



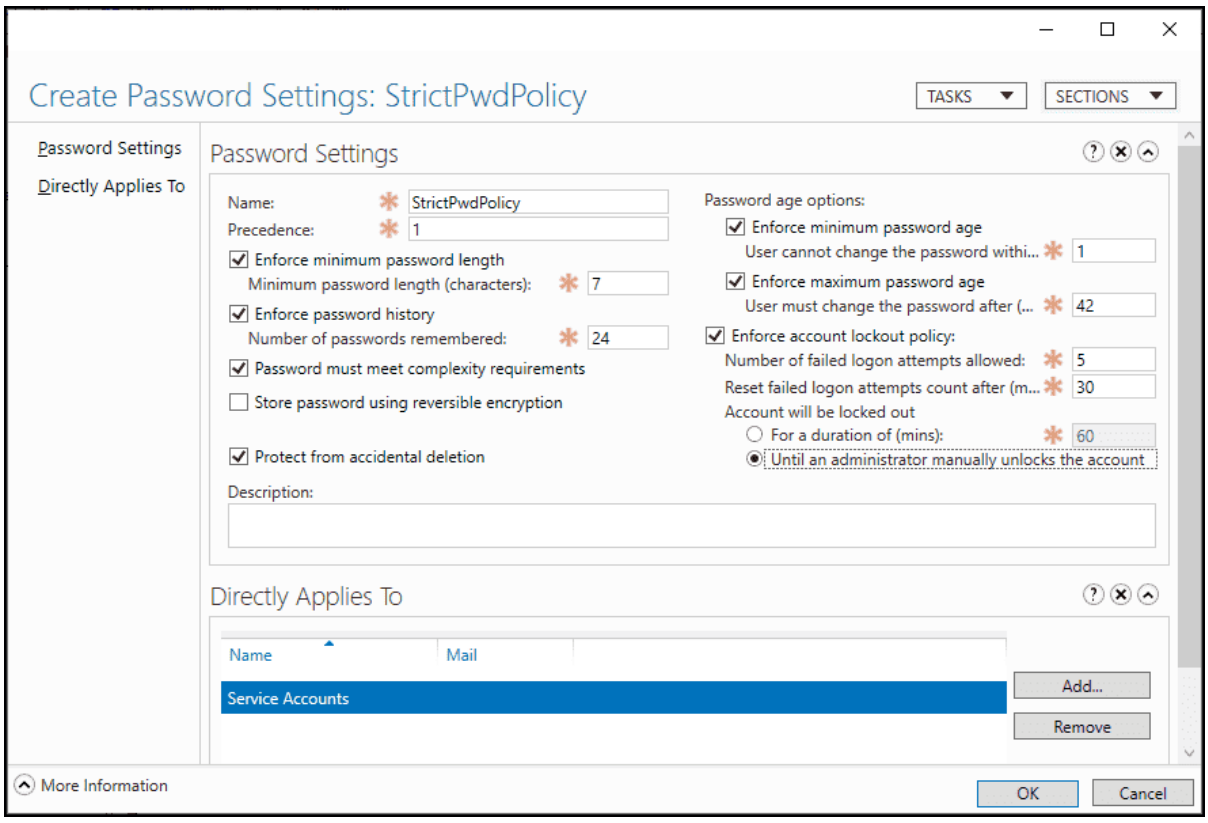
Set a password longer than 127 characters for service accounts using PowerShell

The screenshot shows that I was able to set a password longer than 127 characters for a SQL Server service account with a PowerShell command, but the same command failed when a password longer than 256 characters was tried. The GUI tools wouldn't have allowed setting a password that long.

7. Fine-grained password policy

Back in the days of Server 2003, admins could only define one password policy throughout the domain. This worked fine for some time, but admins later realized the need for different password policies for different sets of users. Windows Server 2008 introduced the concept of a fine-grained password policy, which allows admins to set multiple password policies in a domain.

With the help of fine-grained policies, admins can set up stricter password policies for accounts that are more privileged (e.g., service accounts). The following screenshot shows how to create a new fine-grained password policy using the Active Directory Administrative Center (*dsac*.exe) and directly apply it to a global group.

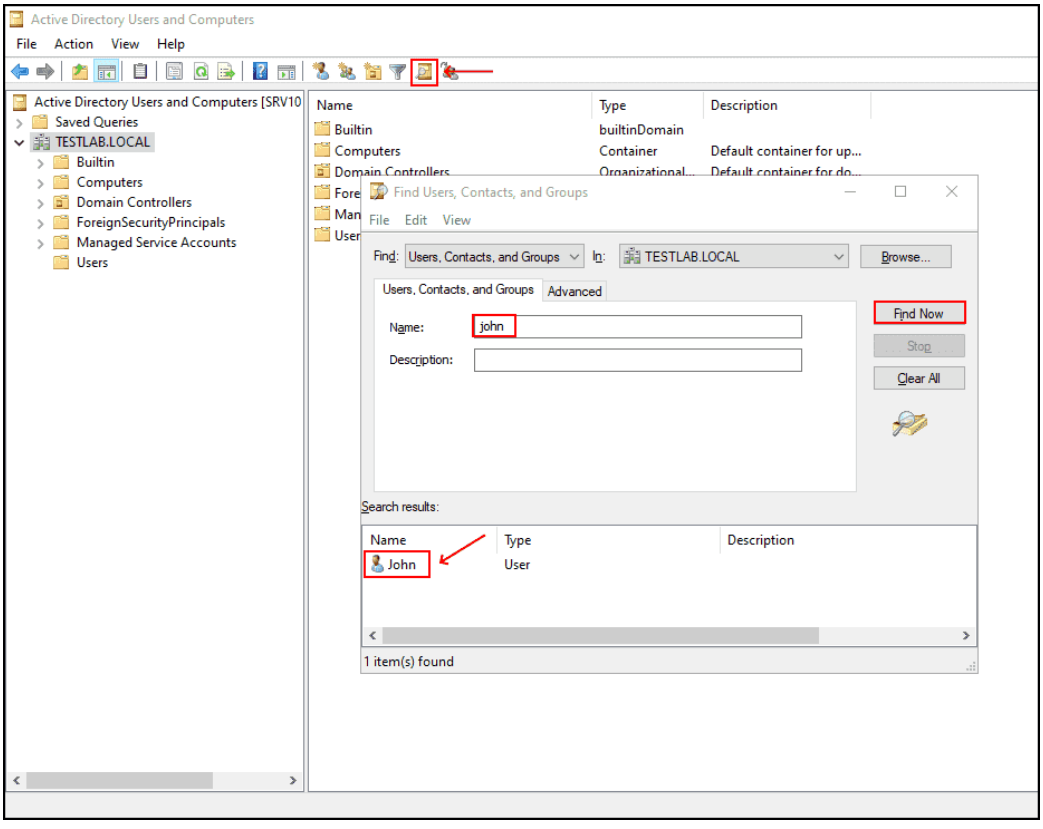


Creating a new fine grained policy using the Active Directory Administrative Center

8. Reset an Active Directory password using the GUI

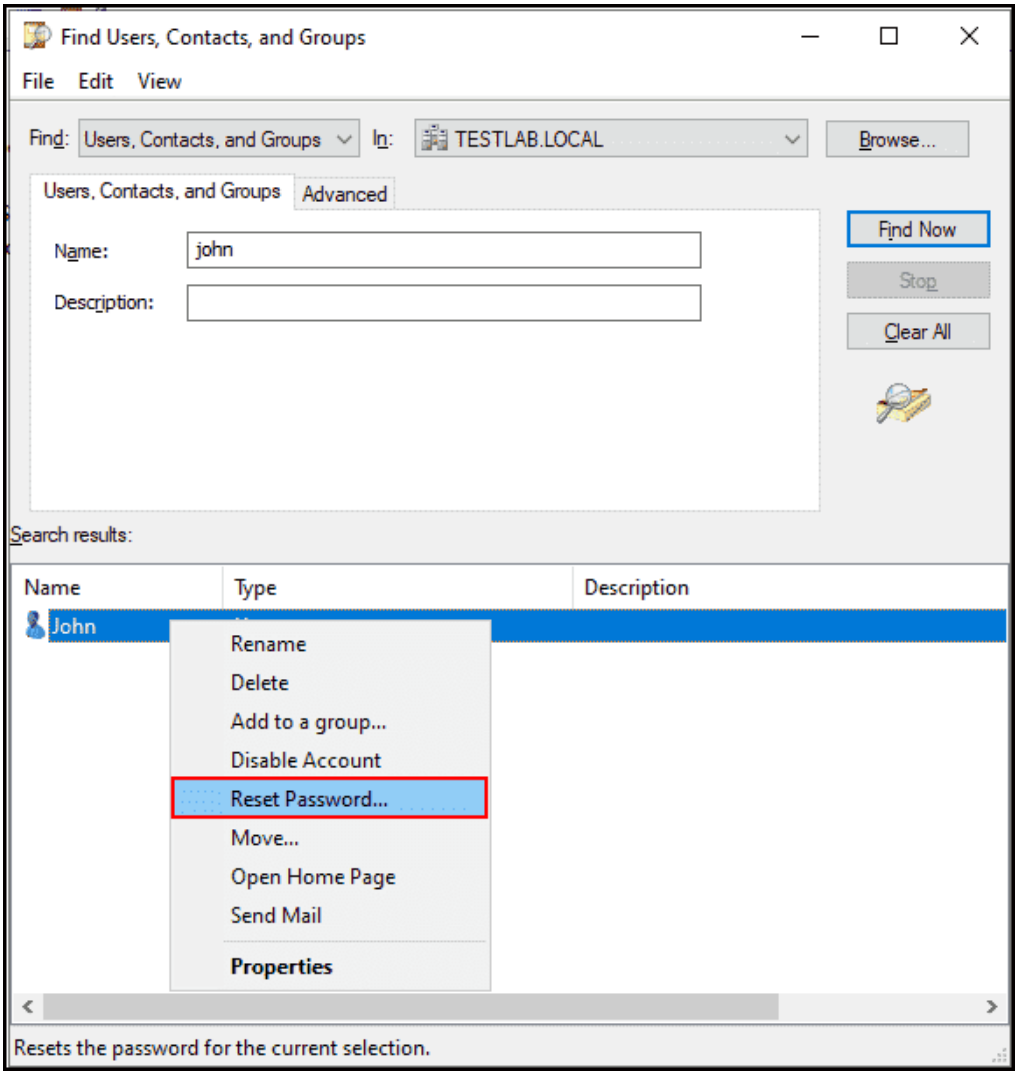
To change a user's password, do the following:

1. Open the Run dialog on any domain controller, type "dsa.msc" without quotes, and press Enter. This will open the Active Directory Users and Computers console.
2. Now, locate the particular user whose password you want to change. You may use the search option if you don't know the exact location of the user.



Searching a user in the Active Directory Users and Computers console

3. Right-click the user and select the *Reset Password* option.



Resetting the AD user password using the GUI tool

4. This will open a Reset Password dialog box. Now type the new password twice and click OK to change it.

9. Reset an Active Directory password using PowerShell

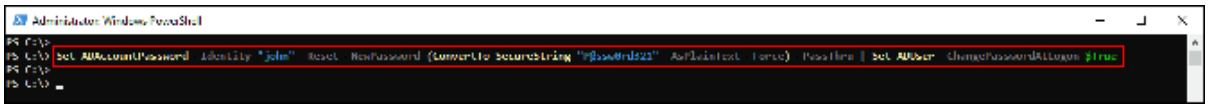
To change the password using PowerShell, do the following:

1. Launch a PowerShell console.
2. Type the following command to open a PS remoting session to one of your domain controllers:

   Enter-PSSession -ComputerName DC1

   Use the *-Credential* parameter to specify alternate credentials if your current user doesn't have sufficient privileges to change AD user passwords. You can skip this step if you've got Active Directory Module for Windows PowerShell installed on your local computer.
3. Now, run the following command to change the AD user password:

   Set-ADAccountPassword -Identity "john" -Reset -NewPassword (ConvertTo-SecureString "P@ssw0rd321" -AsPlainText -Force) -PassThru | Set-ADUser -ChangePasswordAtLogon $True

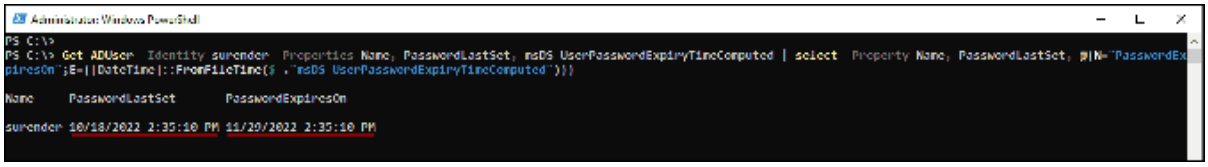


Reset an AD user password using PowerShell

This is a one-line PowerShell command to reset an AD user password. The *-Identity* parameter specifies the username, and the *-NewPassword* parameter specifies a new password as a secure string. We used the *ConvertTo-SecureString* cmdlet to convert a plain-text password to a secure string. Finally, the user object is passed down the pipeline using the *-PassThru* switch parameter, and the *Set-ADUser* cmdlet is used to let users change their password at first login.

To know when the password was last set and when it expires for an AD user, you can use the following PS command:

```
Get-ADUser -Identity surender -Properties Name, PasswordLastSet, msDS-UserPasswordExpiryTimeComputed | select -Property Name, PasswordLastSet, @{N="PasswordExpiresOn";E={[DateTime]::FromFileTime($_."msDS-UserPasswordExpiryTimeComputed")}}
```

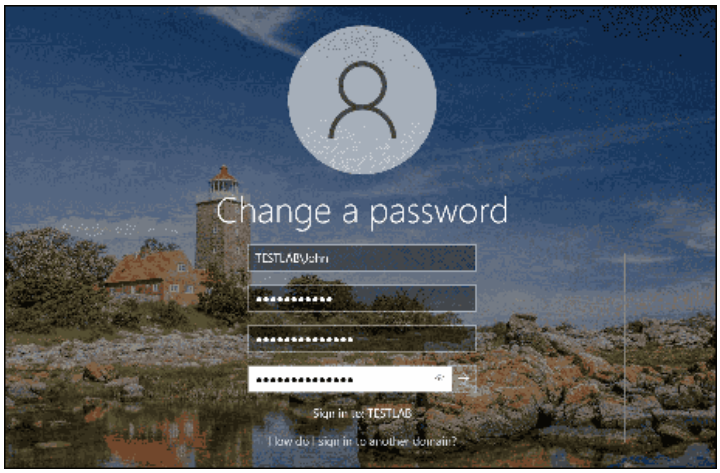


Determine password set date and expiration date for an AD user using PowerShell

Note that *PasswordExpiresOn* isn't a ready attribute, so we used calculated properties to create it.

10. Self-service password update in Active Directory

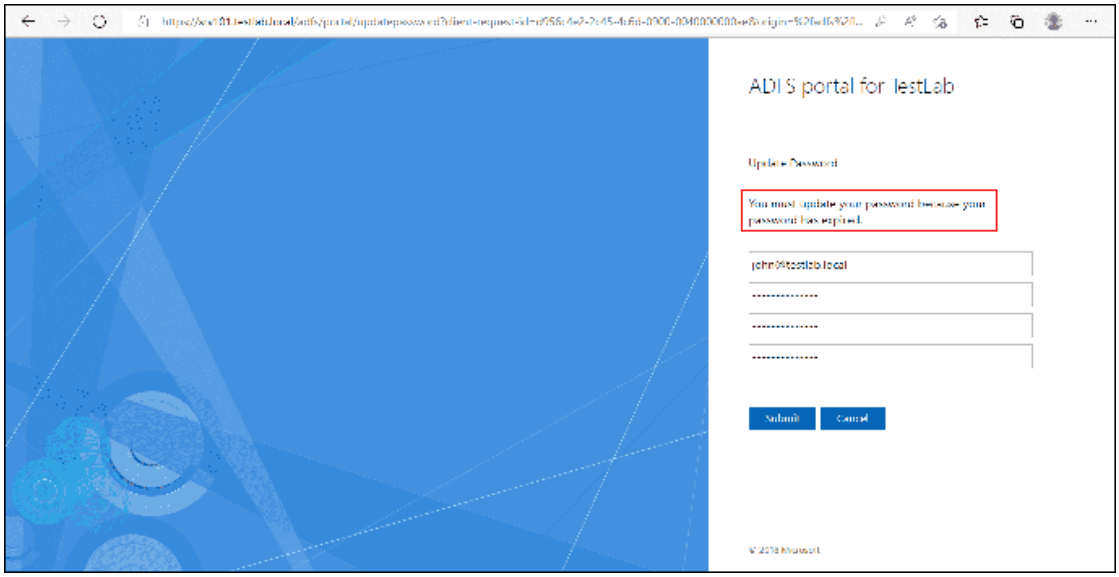
Since Active Directory Domain Service (AD DS) is an enterprise-class service, it isn't always feasible to have admin or helpdesk staff change users' passwords. Users can normally change their password during an interactive login session on a domain-joined machine by using the *Ctrl + Alt + Delete* key combination and then selecting the *Change a password* option.



Change your own password in Windows using Ctrl-Alt-Del

For some situations, this isn't enough, though. For instance, what if a user doesn't have a domain-joined workstation? Maybe they're a remote employee who works from home and uses a VPN to access the corporate network. Or they're a newly hired employee in a remote workforce who just got a temporary password for a new AD account. In either case, the user wouldn't be able to connect to the VPN.

Due to such situations, large-scale enterprises often need a way to enable end users to change their temporary or expired passwords using some sort of self-service portal. Fortunately, the Active Directory Federation Service (ADFS) can be configured to allow this. With the help of the ADFS web portal, users can change their passwords anytime from anywhere. See the following screenshot for reference:



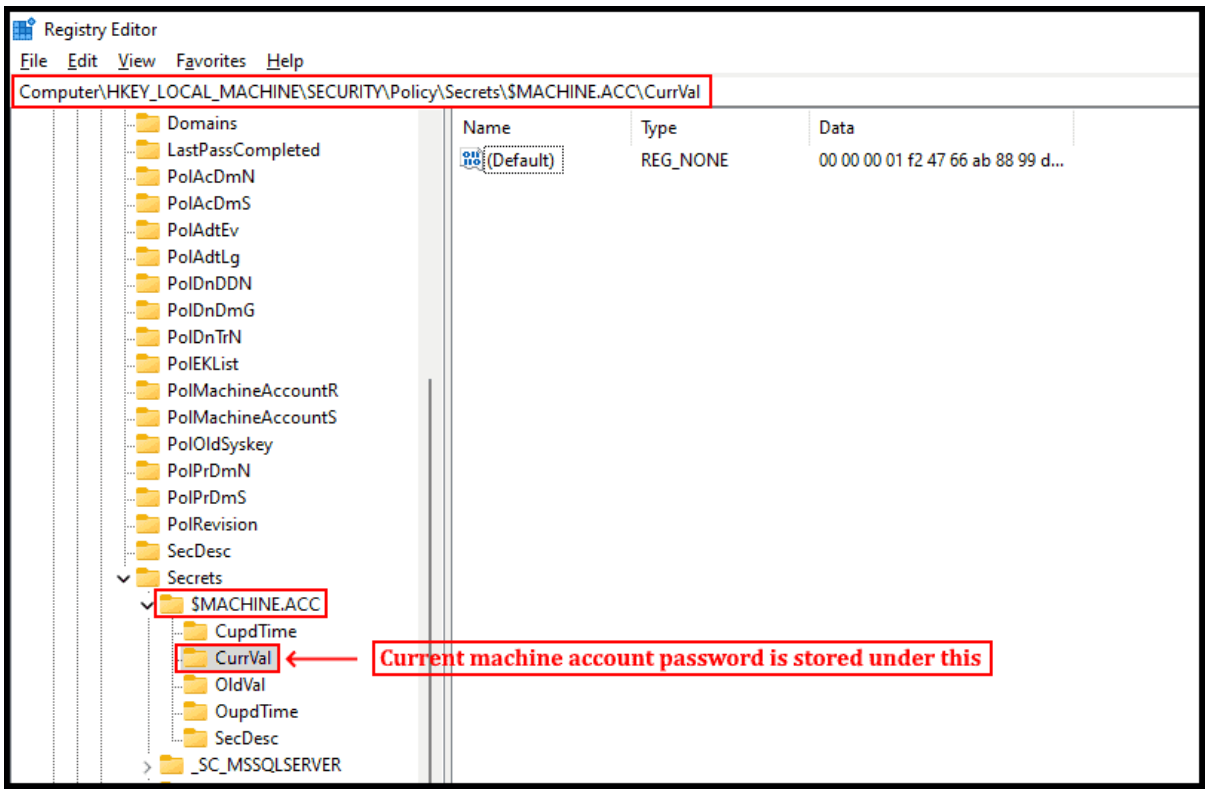
Use the ADFS portal to change a temporary or expired password

11. Computer account password in Active Directory

Just like an AD user account, computer (or machine) accounts also have passwords. Admins do not need to worry about them, since they're managed and changed automatically. This password change is invoked by the client computer itself with the help of the *Netlogon* service. The password is first changed and stored locally and then updated in AD over a secure channel to a domain controller (DC). If the DC refuses this password update, the local change reverts back. On the client computer, the password is locally stored in an encrypted form in the *CurrVal* and *OldVal* subkeys under the *HKEY_LOCAL_MACHINE\SECURITY\Policy\Secrets\$machine.ACC* registry hive.

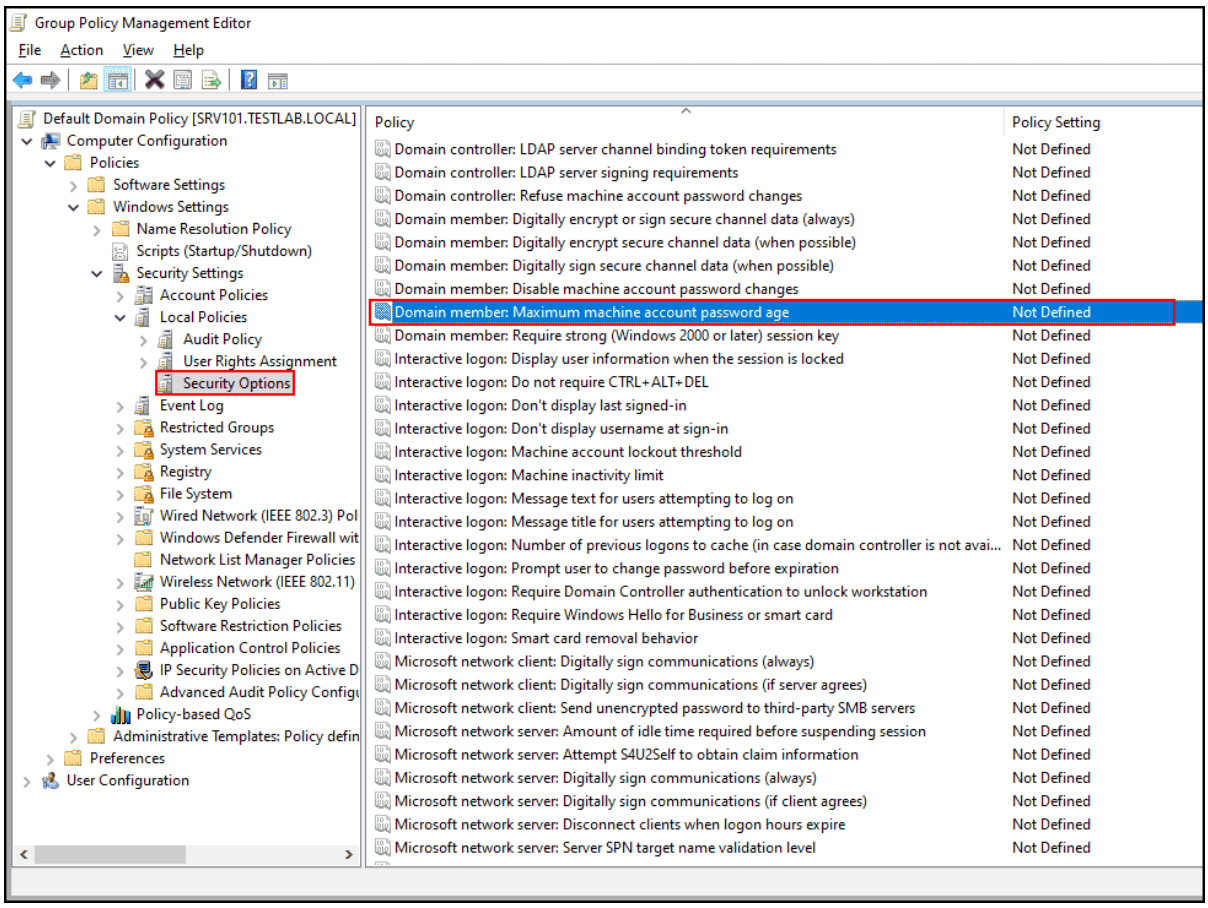
In AD, it is stored under a computer account object in the *unicodepwd* and *lmpwdHistory* attributes. By the way, the local registry hive is accessible to the SYSTEM account only, so if you're interested in exploring it, you can use PsExec to launch the registry editor with SYSTEM account privileges, as shown below:

psexec -i -d -s regedit



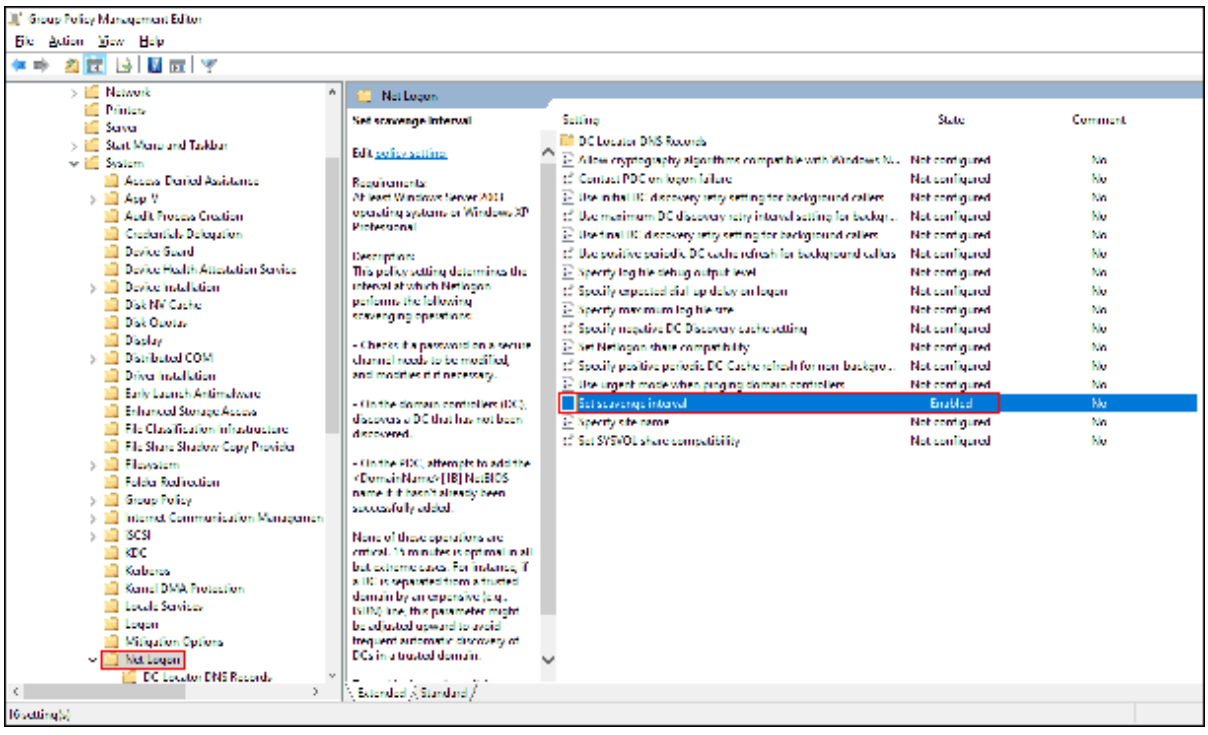
Locating registry hive that stores current machine account passwords

The *Domain member: Maximum machine account password age* group policy setting controls how often the machine account password is changed. It is available under Computer Configuration\Policies\Windows Settings\Security Settings\Local Policies\Security Options.



Domain member Maximum machine account password age group policy setting

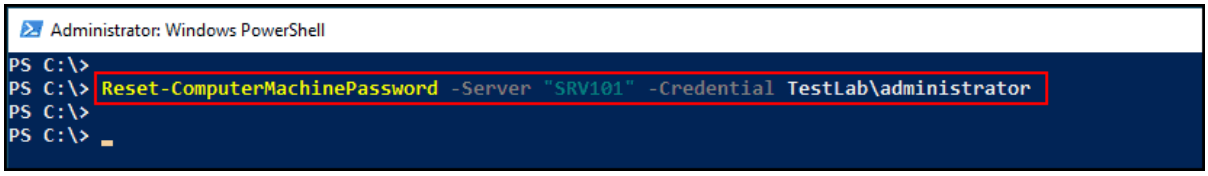
The default value is 30 days, which means the machine account is automatically changed every 30 days, even if the policy is not defined. You can enable this policy and set your preferred number of days. The *Domain member: Disable machine account password changes* group policy is also available to completely disable automatic password reset in a domain, but doing so is not recommended. To initiate the password change, the *Netlogon* service on the client computer invokes a scavenger thread that can be controlled by the *Set scavenge interval* group policy setting available under *Computer Configuration\Policies\Administrative Templates\System\Net Logon.*



Set the scavenge interval group policy setting for Netlogon

Furthermore, you can also reset the machine account password manually using the following PS command on the client computer:

Reset-ComputerMachinePassword -Server "SRV101" -Credential TestLab\administrator

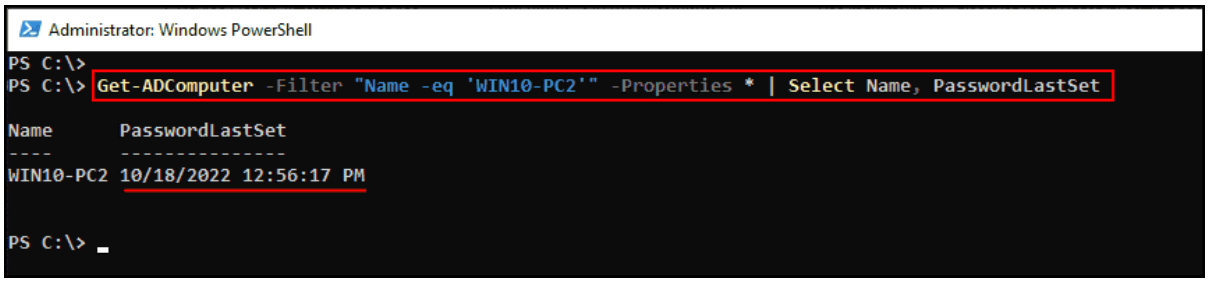


Manually reset the machine account password using PowerShell

The *-Server* parameter is an optional parameter that specifies a preferred DC, and the *-Credential* parameter specifies alternate credentials in case the currently logged-in user doesn't have sufficient privileges. The same command can be used to repair a broken trust relationship between the workstation and the primary domain.

Now, you might be wondering what will happen if the machine account password isn't changed. What happens if the computer remains offline for more than 30 days? Will access to resources be denied? Well, unlike user account passwords, machine account passwords in AD do not expire, since they are exempted from the password policies. However, they are changed automatically every 30 days for security reasons, unless configured otherwise. Therefore, no matter how many days a computer remains offline, it won't be denied access to resources. Moreover, when the computer comes back up, the Netlogon service automatically triggers the password update. To determine when the password was last updated for a computer, you can run the following PS command on a DC:

Get-ADComputer -Filter "Name -eq 'WIN10-PC2'" -Properties * | Select Name, PasswordLastSet



Check when the machine account password was last updated using PowerShell

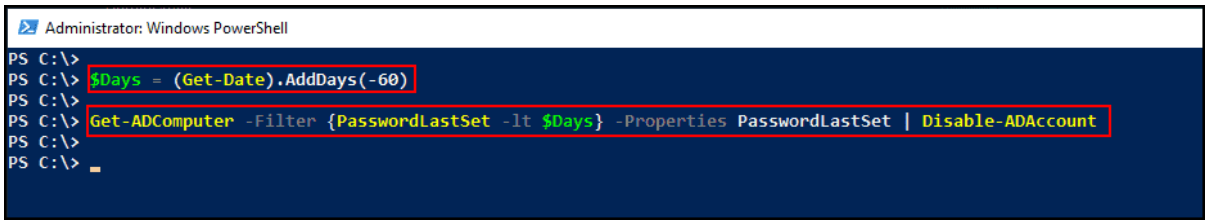
To view the password last set date for all AD computers, run the following command instead:

Get-ADComputer -Filter * -Properties * | Select Name, PasswordLastSet

Some organizations require computer accounts to be disabled if their passwords haven't been changed for a certain number of days. Admins can easily do this with the help of the *PasswordLastSet* attribute, as shown in the following commands:

$Days = (Get-Date).AddDays(-60)

Get-ADComputer -Filter {PasswordLastSet -lt $Days} -Properties PasswordLastSet | Disable-ADAccount



Use PowerShell to disable the AD computers having a password older than a certain number of days

The first command defines the number of days to compare against the *PasswordLastSet* attribute, and the second command disables all AD computers matching the specified filter. Try this command in a test lab environment first.

That was all for this guide. It is true that the IT industry is slowly progressing toward a passwordless approach, but in reality, passwords are not going to vanish completely anytime soon. Therefore, it is a good idea to follow the recommended best practices for password management.