



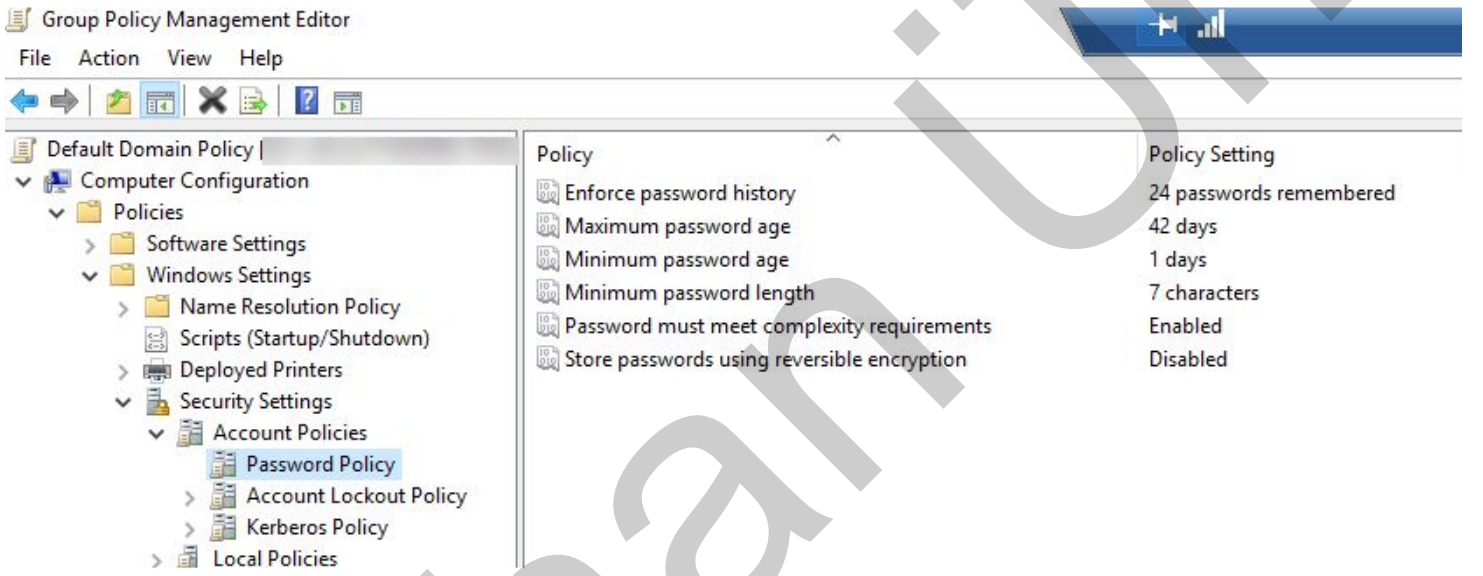
Microsoft

Active Directory

Send email notification before password expired

Active Directory kullanıcı hesaplarına ortamımıza göre güvenlik için Password policyler uyguluyoruz.

Computer Configuration → Windows Settings → Security Settings → Account Policies → Password Policies tıklayıp kendi ortamınıza göre policyi düzenleyebilirsiniz.



Ancak hesabın parolası süresi yaklaştığında kullanıcıya bir email göndermek ve bilgilendirmek güzel olmaz mı? Bence olur.

Bunun için biraz araştırıp aşağıdaki gibi bir powershell scripti buldum. Bunu istediğimiz bir isimde kaydedip çalışmasını istediğimiz bir dizinin altına kopyalamamız gerekiyor. Ben **SifreDegisimBildirimi.ps1** adını verip **Domain Controller**da bir dizin altına kopyaladım.

Konuyla alakalı bir hatırlatma yapayım eğer kullanıcının şifresini kendisinin değiştirmesini istiyorsak bildiğiniz gibi kullanıcı ayarlarındaki **User cannot change password** tikiinin işaretli olmaması gerekiyor. Ayrıca kullanıcıya mail gönderilebilmesi için de yine kullanıcı ayarlarındaki Email attribute kısmının doldurulması gerekiyor.

```
# Domain kullanıcılarına parolalarının süresi dolmadan önce e-posta bildirimleri gönderebilen PowerShell scripti

# AD kullanıcılarını sorgulama
$Users = Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} -Properties DisplayName, EmailAddress, msDS-UserPasswordExpiryTimeComputed | `
Select-Object -Property DisplayName, EmailAddress, @{Name="ExpirationDate";Expression={[datetime]::FromFileTime($_.msDS-UserPasswordExpiryTimeComputed)}} | `
Sort-Object "ExpirationDate"

# Parolanın süresinin yakında dolup dolmayacağını kontrol edip ve e-posta bildirimi göndermek.
$UserList = foreach ($User in $Users) {
    if ($User.ExpirationDate -le (Get-Date).AddDays(7) -and $User.ExpirationDate -ge (Get-Date))
    {
        # Parolası yakında sona erecek kullanıcıların listesini kaydetmek için PSCustomObject oluşturmak
        [PSCustomObject]@{
            Name = $User.DisplayName
            EmailAddress = $User.EmailAddress
            ExpiryDate = $User.ExpirationDate
        }
        # Email gönderimi
        [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
        $SMTP = "smtp-mail.outlook.com"
        $From = "mail@domain.com"
        $Username = "mail@domain.com"
        #Read-Host -Prompt "Mail şifresini giriniz: " -AsSecureString | ConvertFrom-SecureString | Out-File "C:\MailLog\cred.txt"
        $Pass = Get-Content "C:\MailLog\cred.txt" | ConvertTo-SecureString
        $Cred = New-Object System.Management.Automation.PSCredential -argumentlist $Username, $Pass
        $Subject = "Oturum Acma Sifrenizin Süresi Yakında Dolmak Üzere!!!"
        $Body = "Sayın $($User.DisplayName), " + "`n`nOturum acma sifrenizin süresi $User.ExpirationDate tarihinde dolacaktır. Lütfen yakın zamanda değiştiriniz." + "`n`nSaygılarımızla..," + "`nIT Departmanı"
        $EmailBody = "Merhaba $($User.DisplayName), " + "`n`n" +
            "Bu, sifrenizin $($User.ExpirationDate) tarihinde sona ereceğini hatırlatmak içindir." + "`n`n" +
            "Hesabınıza erişim sağlayabilmek için lütfen sifrenizi değiştiriniz." + "`n`n" +
            "Teşekkürler," + "`n`n" +
            "IT Departmanı"

        # E-postayı göndermeye çalışma, istisnaları yakalama ve günlüğe kaydetme
        # Send-MailMessage -From $From -To $User.EmailAddress -Subject $Subject -Body $EmailBody -smtpserver $SMTP -usesssl -Credential $cred -Port 587
        $EmailMessage = @{
            From = $From
            To = $User.EmailAddress
            Subject = $Subject
            Body = $EmailBody
            SmtpServer = $SMTP
            UseSsl = $True
            Credential = $cred
            Port = 587
        }
        Send-MailMessage @EmailMessage
        # Logları dosyaya yazdır
        Add-Content -Path "C:\MailLog\EmailSuccess.log" -Value "Email başarıyla gönderildi. Kullanıcı : $($User.DisplayName) - Email : $($User.EmailAddress) - Gönderim Tarihi : $(Get-Date)"
    }
    Catch {
        # Logları dosyaya yazdır.
        Write-Host "Email gönderilemedi. Kullanıcı : $($User.DisplayName) - Email : $($User.EmailAddress): $_"
        Add-Content -Path "C:\MailLog\EmailFailure.log" -Value "Email gönderilemedi. Kullanıcı : $($User.DisplayName) - Email : $($User.EmailAddress) Tarih : $(Get-Date) $_"
    }
}
$UserList | sort-object ExpirationDate
```

Script aşağıdaki gibi, kırmızı alanları kendi bilgileriniz ile doldurabilirsiniz. Ben son 7 günü kalan kullanıcılar için günlük hatırlatma yapılmasını istiyorum.

Domain kullanıcılarına parolalarının süresi dolmadan önce e-posta bildirimleri gönderebilen PowerShell scripti

AD kullanıcılarını sorgulama

```
$Users = Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} -Properties DisplayName,
EmailAddress, msDS-UserPasswordExpiryTimeComputed | `
Select-Object -Property DisplayName, EmailAddress,
@{Name="ExpirationDate";Expression={[datetime]::FromFileTime($_.msDS-UserPasswordExpiryTimeComputed)}} | `
Sort-Object "ExpirationDate"
```

Parolanın süresinin yakında dolup dolmayacağını kontrol edip ve e-posta bildirimi göndermek.

```
$UserList = foreach ($User in $Users) {
    if ($User.ExpirationDate -le (Get-Date).AddDays(7) -and $User.ExpirationDate -ge (Get-Date))
    {
```

```
        # Parolası yakında sona erecek kullanıcıların listesini kaydetmek için PSCustomObject oluşturmak
        [PSCustomObject]@{
            Name = $User.DisplayName
            EmailAddress = $User.EmailAddress
            ExpiryDate = $User.ExpirationDate
        }
```

Email gönderimi

```
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
$SMTP = "smtp-mail.outlook.com"
$From = "mail@domain.com"
$Username = "mail@domain.com"
```

```
#Read-Host -Prompt "Mail şifresini giriniz: " -AsSecureString | ConvertFrom-SecureString | Out-File
"C:\MailLog\cred.txt"
```

```
$Pass = Get-Content "C:\MailLog\cred.txt" | ConvertTo-SecureString
```

```
$Cred = New-Object System.Management.Automation.PSCredential -argumentlist $Username, $Pass
```

```
$Subject = "Oturum Acma Sifrenizin Süresi Yakında Dolmak Üzere!!!"
```

```
$Body = "Sayın $($User.DisplayName), " + "`n`nOturum acma sifrenizin süresi $User.ExpirationDate tarihinde
dolacaktır. Lütfen yakın zamanda değiştiriniz." + "`n`nSaygılarımızla..," + "`nIT Departmanı"
```

```
$EmailBody = "Merhaba $($User.DisplayName), " + "`n`n" +
```

"Bu, sifrenizin \$(\$User.ExpirationDate) tarihinde sona erecegini hatirlatmak icindir." + "`n`n" +
"Hesabiniza erisim saglayabilmek icin lutfen sifrenizi degistiriniz." + "`n`n" +
"Tesekkurler," + "`n" +
"IT Departmani"

E-postayı göndermeye çalışma, istisnaları yakalama ve günlüğe kaydetme

Try {

Send-MailMessage -From \$From -To \$User.EmailAddress -Subject \$Subject -Body \$EmailBody -smtpserver \$SMTP -
usessl -Credential \$cred -Port 587

\$EmailMessage = @{

From = \$From

To = \$User.EmailAddress

Subject = \$Subject

Body = \$EmailBody

SmtpServer = \$SMTP

UseSsl = \$True

Credential = \$cred

Port = 587

}

Send-MailMessage @EmailMessage

Logları dosyaya yazdir

Add-Content -Path "C:\MailLog\EmailSuccess.log" -Value "Email basariyla gonderildi. Kullanici :
\$(\$User.DisplayName) - Email : \$(\$User.EmailAddress) - Gonderim Tarihi : \$(Get-Date) "

}

Catch {

Logları dosyaya yazdir.

Write-Host "Email gonderilemedi. Kullanici : \$(\$User.DisplayName) - Email : \$(\$User.EmailAddress): \$_"

Add-Content -Path "C:\MailLog\EmailFailure.Log" -Value "Email gonderilemedi. Kullanici : \$(\$User.DisplayName) -
Email : \$(\$User.EmailAddress) Tarih : \$(Get-Date) \$_"

}

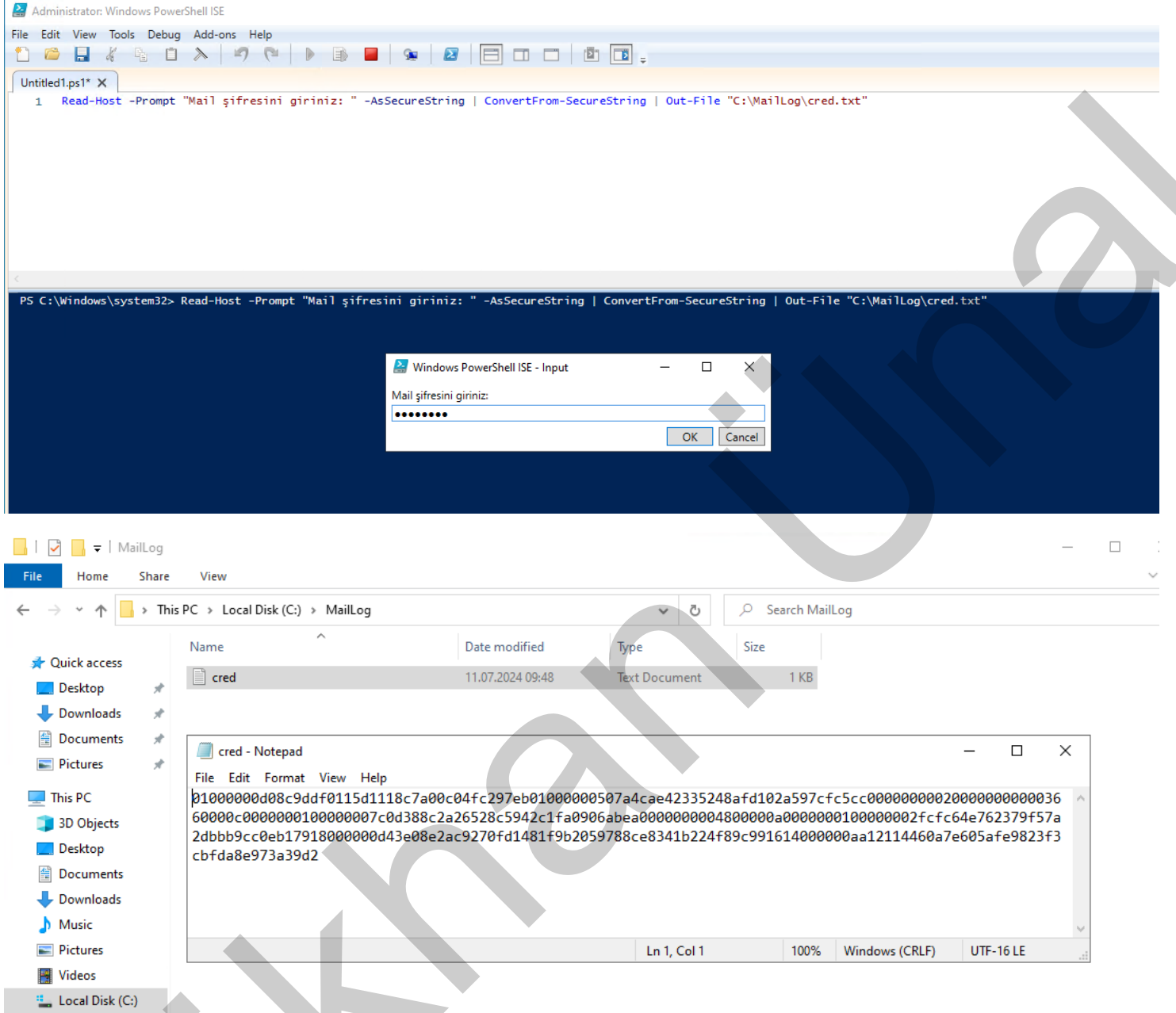
}

}

\$Userlist | sort-object ExpirationDate

Scriptte yeşil ile işaretlediğim kısım bir kere çalıştırılacak.

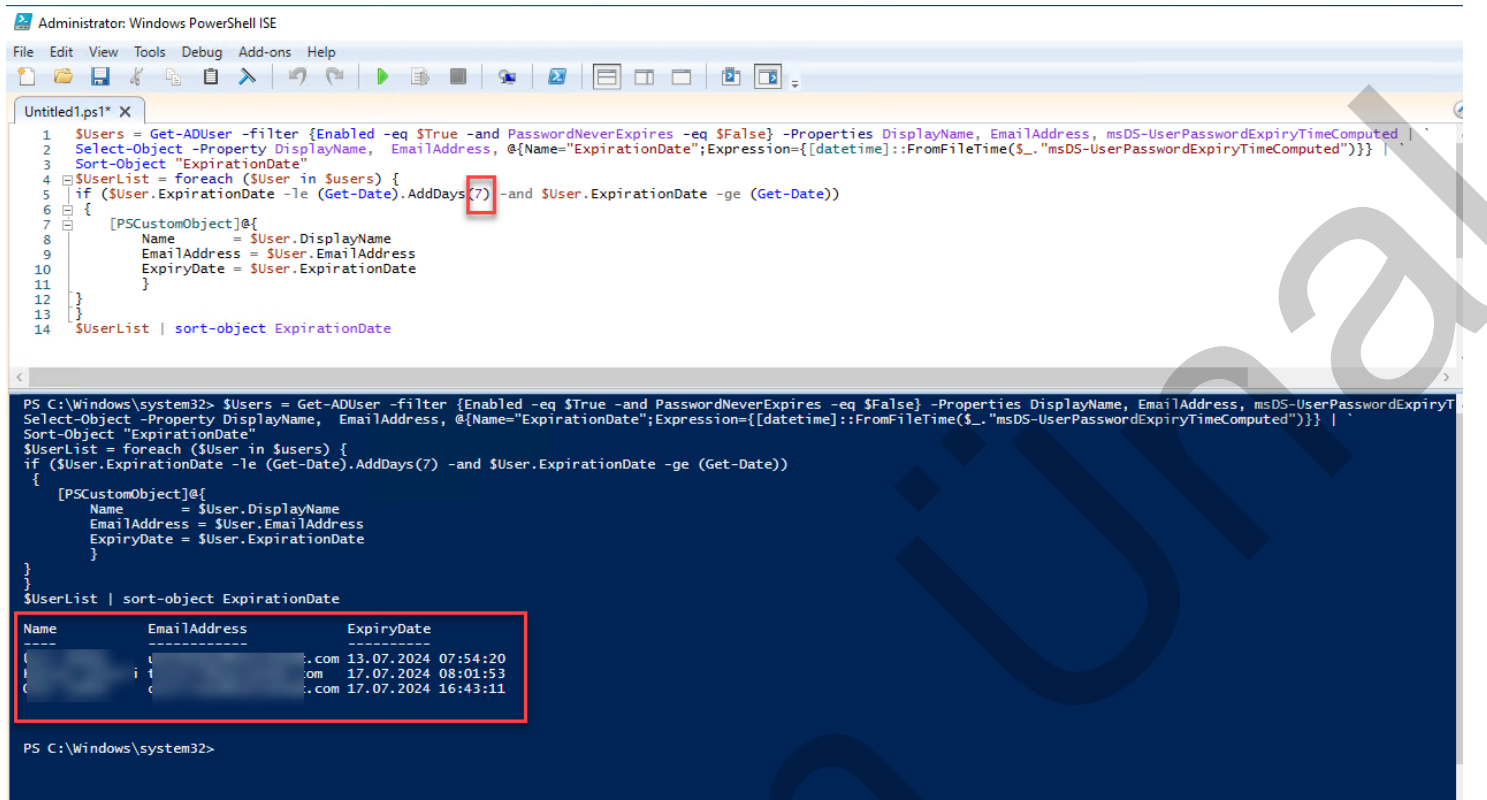
Daha önce paylaştığım yazımda Pentestcilerin sevdiği bir durum olduğu için bu sefer birazda olsa güvenli olması açısından gönderim yapacak mail şifremizi hashleyeceğiz.



Bu komutu şifre değişmediği sürece tekrar kullanmayacağız zaten scriptte de # ile yorum satırı halinde.

Şimdi de 7 gün içinde şifresi dolacak kullanıcımız var mı bir kontrol edelim. (Bu kısım script içinde var sadece makale için test amacıyla yapıldı. Zorunlu değil)

Görüldüğü gibi 3 kullanıcının süresi dolmak üzere.



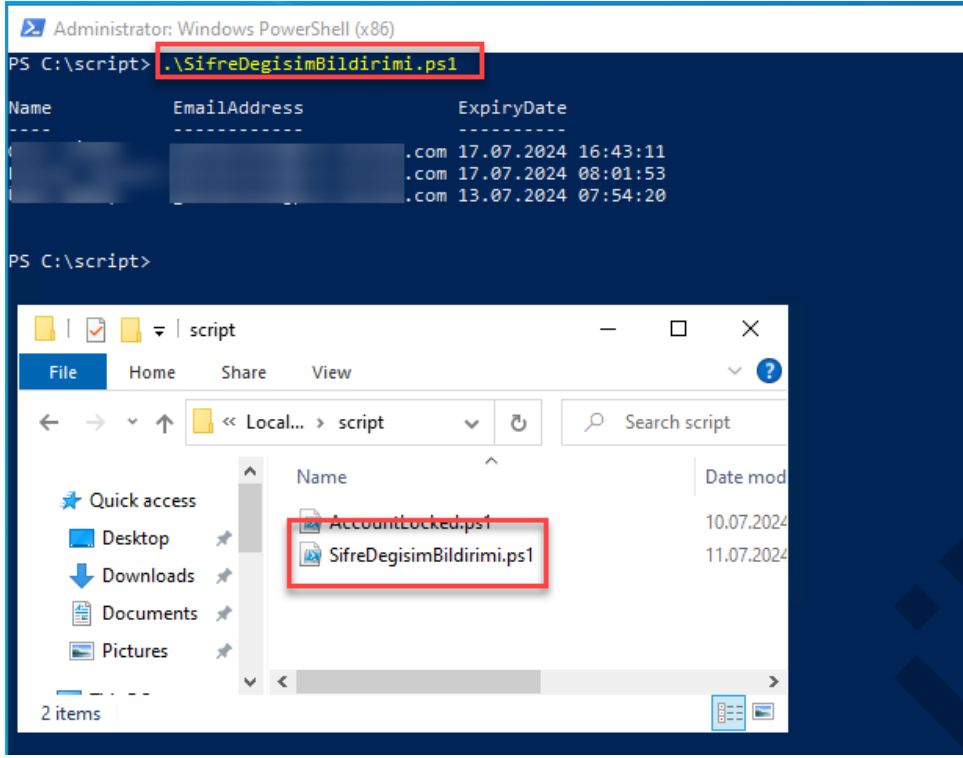
```
1 $Users = Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} -Properties DisplayName, EmailAddress, msDS-UserPasswordExpiryTimeComputed | `
2 Select-Object -Property DisplayName, EmailAddress, @{Name="ExpirationDate";Expression={[datetime]::FromFileTime($_.msDS-UserPasswordExpiryTimeComputed)}} | `
3 Sort-Object "ExpirationDate"
4 $UserList = foreach ($User in $Users) {
5     if ($User.ExpirationDate -le (Get-Date).AddDays(7) -and $User.ExpirationDate -ge (Get-Date))
6     {
7         [PSCustomObject]@{
8             Name = $User.DisplayName
9             EmailAddress = $User.EmailAddress
10            ExpiryDate = $User.ExpirationDate
11        }
12    }
13 }
14 $UserList | sort-object ExpirationDate
```

```
PS C:\Windows\system32> $Users = Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} -Properties DisplayName, EmailAddress, msDS-UserPasswordExpiryTimeComputed | `
Select-Object -Property DisplayName, EmailAddress, @{Name="ExpirationDate";Expression={[datetime]::FromFileTime($_.msDS-UserPasswordExpiryTimeComputed)}} | `
Sort-Object "ExpirationDate"
$UserList = foreach ($User in $Users) {
    if ($User.ExpirationDate -le (Get-Date).AddDays(7) -and $User.ExpirationDate -ge (Get-Date))
    {
        [PSCustomObject]@{
            Name = $User.DisplayName
            EmailAddress = $User.EmailAddress
            ExpiryDate = $User.ExpirationDate
        }
    }
}
$UserList | sort-object ExpirationDate
```

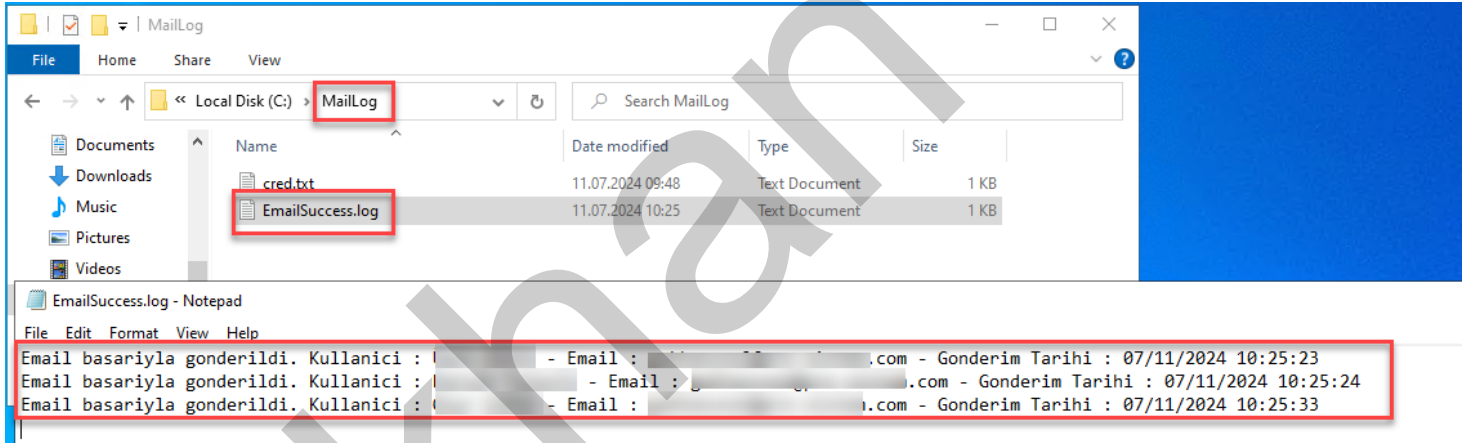
Name	EmailAddress	ExpiryDate
i t .com		13.07.2024 07:54:20
i t .com		17.07.2024 08:01:53
c .com		17.07.2024 16:43:11

```
$Users = Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} -Properties DisplayName,
EmailAddress, msDS-UserPasswordExpiryTimeComputed | `
Select-Object -Property DisplayName, EmailAddress,
@{Name="ExpirationDate";Expression={[datetime]::FromFileTime($_.msDS-UserPasswordExpiryTimeComputed)}} | `
Sort-Object "ExpirationDate"
$UserList = foreach ($User in $Users) {
if ($User.ExpirationDate -le (Get-Date).AddDays(7) -and $User.ExpirationDate -ge (Get-Date))
{
[PSCustomObject]@{
    Name = $User.DisplayName
    EmailAddress = $User.EmailAddress
    ExpiryDate = $User.ExpirationDate
}
}
}
$UserList | sort-object ExpirationDate
```

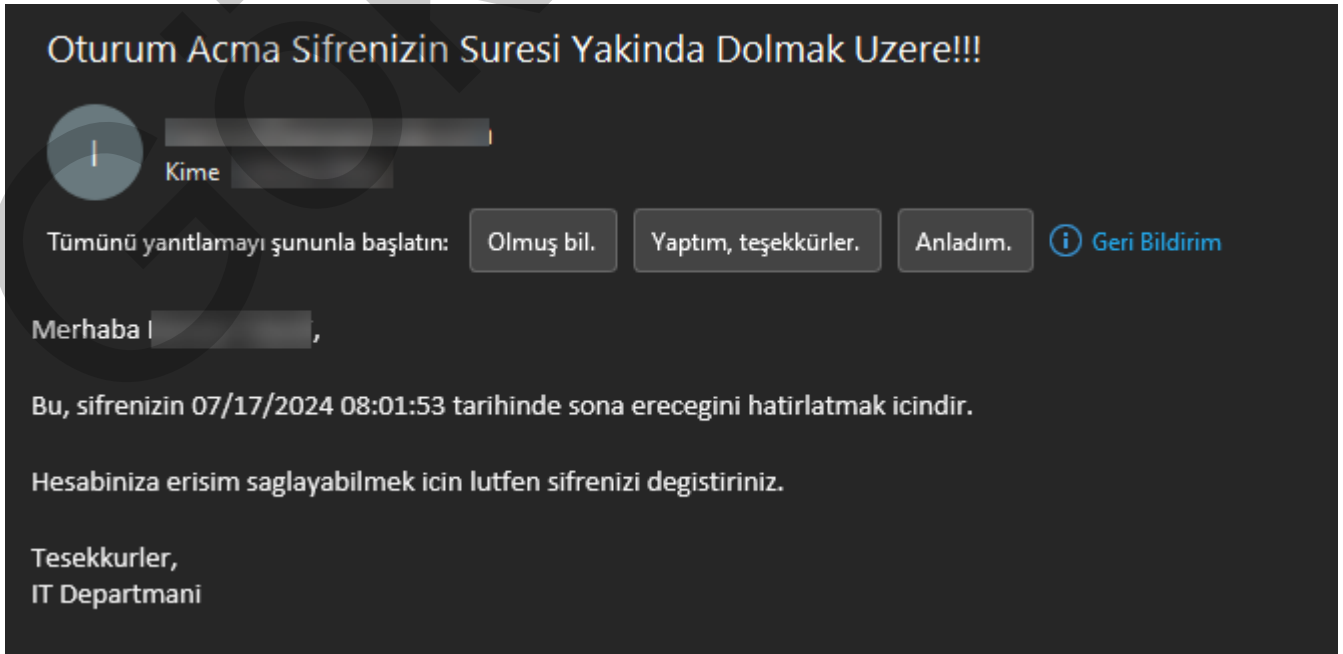
Artık scriptimi test ediyorum. Powershell açıp çalıştırıyorum.



Script çalıştı ve mail gönderim logunu oluşturdu.



Email olarakta kullanıcıya bildirim gönderildi.



Bundan sonraki işlem için yine **Domain Controller** da **Task Scheduler** çalıştırıyorum ve Yeni bir task oluşturuyorum. **General** sekmesinde aşağıdaki gibi isim veriyorum. Hangi yetkili kullanıcı ile çalıştıracağımı seçiyorum. Ardından oturum açık olsa da olmasa da çalışmasını sağlamak için “**Run whether user is logged on or not**” işaretliyorum.

Create Task

General Triggers Actions Conditions Settings

Name: SifreDegisimBildirimEmail

Location: \Microsoft\Windows

Author:

Description:

Security options

When running the task, use the following user account:

Change User or Group...

☐ Run only when user is logged on

☒ Run whether user is logged on or not

☒ Do not store password. The task will only have access to local computer resources.

☒ Run with highest privileges

☐ Hidden

Configure for: Windows Vista™, Windows Server™ 2008

OK Cancel

Trigger sekmesine gelip **New Trigger** tıklıyorum. Daily seçiyorum ve çalışmasını istediğim saati belirleyip Ok diyerek kaydediyorum.

Create Task

General Triggers Actions Conditions Settings

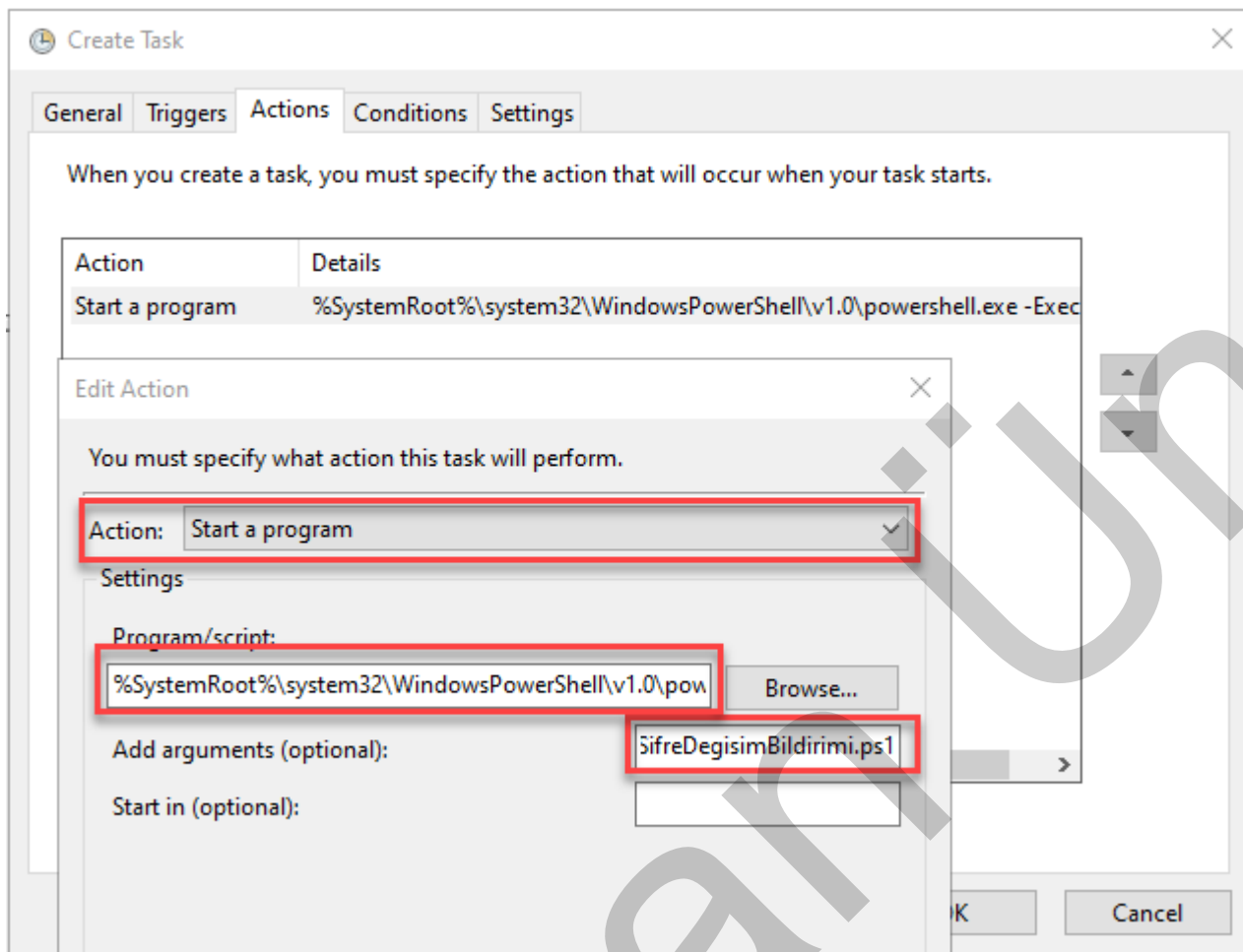
When you create a task, you can specify the conditions that will trigger the task.

Trigger	Details	Status
Daily	At 09:00 every day	Enabled

New... Edit... Delete

OK Cancel

Ardından **Action** sekmesine gelip **New Action** tıklıyorum. **Action : Start a program** seçiyorum. Program script olarak powershell.exe dosyasının yolunu gösteriyorum. “%SystemRoot%\system32\WindowsPowerShell\v1.0\powershell.exe” Add argument olarak çalıştıracığım powershell dosyasının yolunu gösteriyorum. “-ExecutionPolicy Bypass -File C:\MailLog\SifreDegisimBildirimi.ps1” ” Ok ile kaydediyorum.



Tekrar OK ile kaydediyorum. Görevim hazır hale geldi.

Name	Status	Triggers	Next Run Time
Server Initial Configuration Task	Disabled	At system startup	
SifreDegisimBildirimiEmail	Ready	At 09:00 every day	12.07.2024 09:00:00

İşlemler bu kadar. Zaten scripti çalıştırdığımızda mail bildirimi geldiğini ve log dosyası oluştuğunu görmüştük. Oluşturduğumuz Task Schedule zamanı geldiğinde script çalışacaktır.