# Using Group Managed Service Accounts

## What/Why?

A **gMSA** (Group Managed Service Account) is a special type of account in Windows Server designed to run services securely and efficiently across multiple servers without requiring administrators to manage passwords manually.

It is an extension of **sMSA** (Standalone Managed Service Accounts), which only work on a single server. I recommend always using gMSAs because they are more versatile.

## How to set this up?

There are a few scripts in this manual, you need to use run those as administrator. On the DC and on the member servers

### Domaincontroller

A gMSA account can only be created through powershell. This is the part you need to do on the domain controller.

```
$ServiceName = Read-Host -Prompt "Give the name of the service account (gMSA_
will be added: "
$Description = Read-Host -Prompt "Give a description for this account: "
$DestServerName = Read-Host -Prompt "What is the server name where you want to
use the gMSA account: "

#region Domain Controller

#gMSA name and hosts using this gMSA
$gMSAName = "gMSA_$ServiceName" #gMSA Name (if change here must also change below
in second part of script that is run on Probe server - line ~81)
$gMSAGroupName = "gMSAG_$ServiceName" #gMSA group name
$gMSADescription = "Service Account for $Description"

#Needs KDS Root Key to be generated first if not already done for this Domain

$KeyExists = Get-KdsRootKey
if($KeyExists){
    $KeyCount = $KeyExists.Count
    Write-Host "$KeyCount x KDS Root Key exists - No need to create" -
ForegroundColor Green
    if($KeyCount -gt 1){
        Write-Host "$KeyCount x KDS Root Keys - Check if multiple needed" -
ForegroundColor Yellow
        $KeyExists
    }
} else {
    Write-Host "KDS Root Key does not exist - Creating key" -ForegroundColor
Yellow
    Add-KdsRootKey -EffectiveTime ((get-date).addhours(-10))
}
#Refer https://learn.microsoft.com/en-us/powershell/module/kds/add-
kdsrootkey?view=windowsserver2022-ps
#And https://learn.microsoft.com/en-us/answers/questions/441587/i-ran-add-
kdsrootkey-effectiveimmediately-and-now
```

```
#Get the domain details
$MSAUserServer = Get-ADComputer $DestServerName
$ADDomainDetails = Get-ADDomain
$ADDNSRoot = $ADDomainDetails.DNSRoot
$ADDistName = $ADDomainDetails.DistinguishedName

#Create the group that contains the hosts that can use this gMSA
New-ADGroup -Name $gMSAGroupName -GroupCategory Security -GroupScope Global -Path
"CN=Managed Service Accounts,$ADDistName" -Description "Group for use by gMSA
$gMSAName"

#Create the gMSA
New-ADServiceAccount -Name $gMSAName -Description $gMSADescription -DNSHostName
"$gMSAName.$ADDNSRoot" -ManagedPasswordIntervalInDays 10 -
PrincipalsAllowedToRetrieveManagedPassword $gMSAGroupName -Enabled $True

if(!$error){
    Write-Host "$gMSAName created and $gMSAGroupName group created" -
ForegroundColor Green
}
if ($gMSAName[-1] -ne "$") { $gMSAName += "$" }

Test-AdServiceAccount $gMSAName
```

Add the server(s) that will use the gMSA accounts to the group gMSAG_*[ServiceNaam]*

## Service

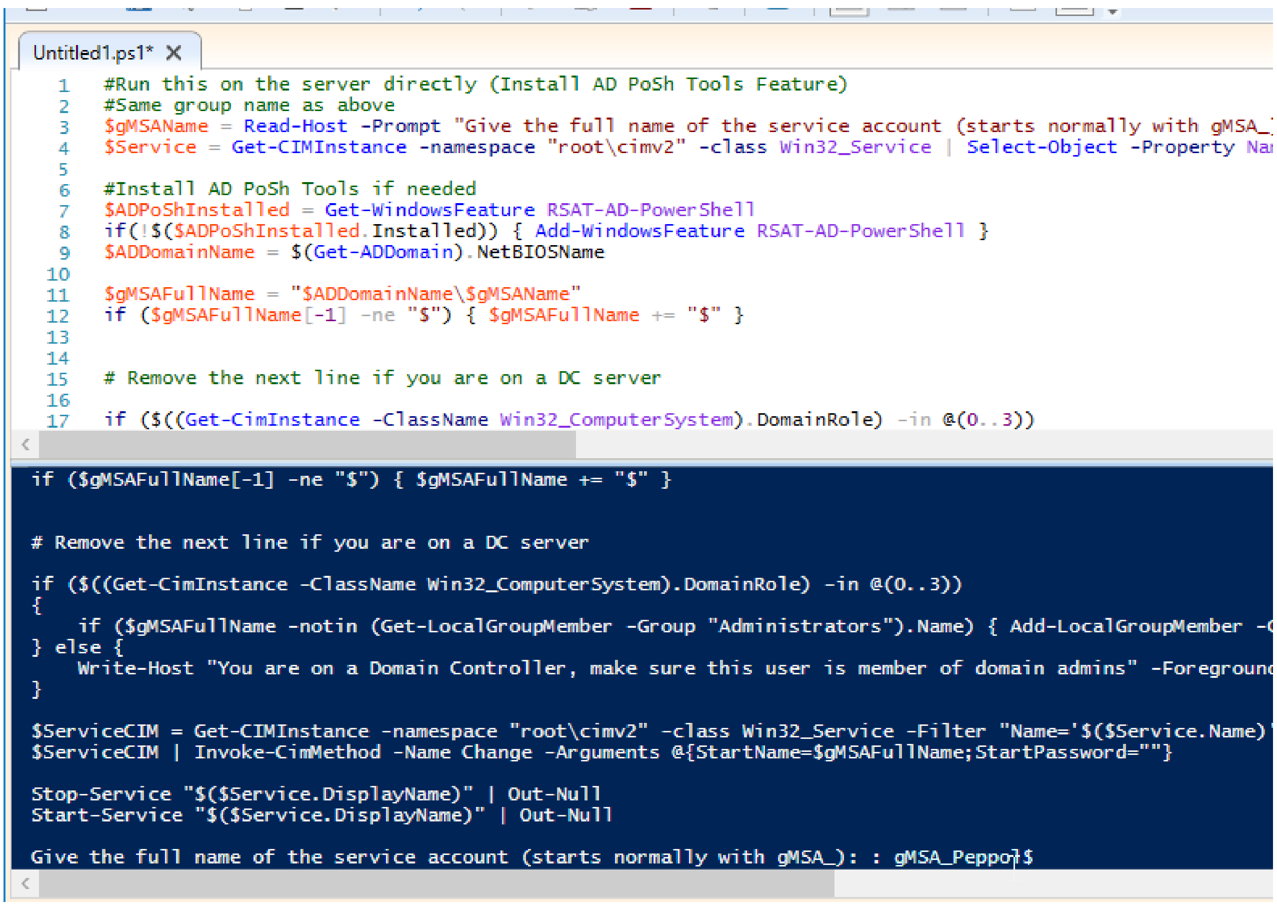
When using this for an N-able probe. If a probe is already installed, you don't need to do something special, if you need to setup a new probe. Don't take the detour with a setup with a domain account, but install it as local service, and change it with this script to a gMSA account.

By default gMSA accounts are used for services. This is the powershell script you can run on a member server to change a service to a gMSA..

```
#Run this on the server directly (Install AD PoSh Tools Feature)
#Same group name as above
$gMSAName = Read-Host -Prompt "Give the full name of the service account (starts
normally with gMSA_): "
$Service = Get-CIMInstance -namespace "root\cimv2" -class Win32_Service | Select-
Object -Property Name,DisplayName,State,StartMode,StartName | Out-GridView -
PassThru -Title "Choose the service to change to this gMSA account"

#Install AD PoSh Tools if needed
$ADPoShInstalled = Get-WindowsFeature RSAT-AD-PowerShell
if(!$($ADPoShInstalled.Installed)) { Add-WindowsFeature RSAT-AD-PowerShell }
$ADDomainName = $(Get-ADDomain).NetBIOSName

$gMSAFullName = "$ADDomainName\$gMSAName"
if ($gMSAFullName[-1] -ne "$") { $gMSAFullName += "$" }


# Remove the next line if you are on a DC server
if ($((Get-CimInstance -ClassName Win32_ComputerSystem).DomainRole) -in @(0..3))
{
    if ($gMSAFullName -notin (Get-LocalGroupMember -Group "Administrators").Name)
{ Add-LocalGroupMember -Group "Administrators" -Member $gMSAFullName }
} else {
    Write-Host "You are on a Domain Controller, make sure this user is member of
domain admins" -ForegroundColor Yellow
}

$ServiceCIM = Get-CIMInstance -namespace "root\cimv2" -class Win32_Service -
Filter "Name='$($Service.Name)'"
$ServiceCIM | Invoke-CimMethod -Name Change -Arguments
@{StartName=$gMSAFullName;StartPassword=""}

Stop-Service "$($Service.DisplayName)" | Out-Null
Start-Service "$($Service.DisplayName)" | Out-Null
```

This script asks 2 things:

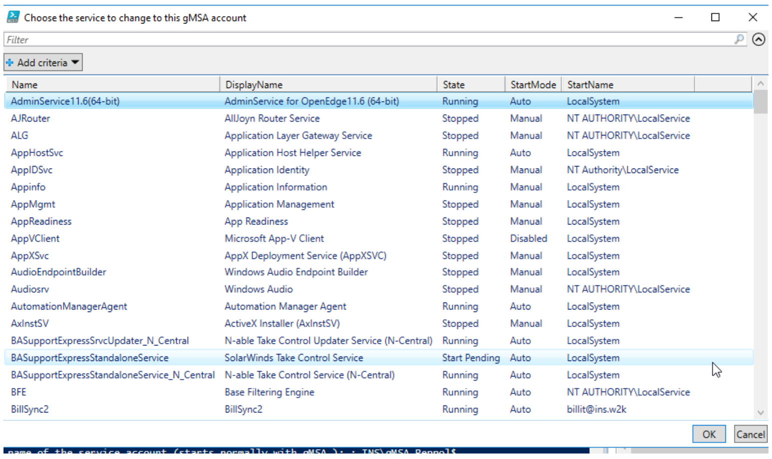
1. It asks the name of the gMSA account in the form gmsa_xxx (without domain\ in front of it)



```
#Run this on the server directly (Install AD PoSh Tools Feature)
#Same group name as above
$gMSAName = Read-Host -Prompt "Give the full name of the service account (starts normally with gMSA_]
$Service = Get-CIMInstance -namespace "root\cimv2" -class Win32_Service | Select-Object -Property Nai

#Install AD PoSh Tools if needed
$ADPoShInstalled = Get-WindowsFeature RSAT-AD-PowerShell
if(!$($ADPoShInstalled.Installed)) { Add-WindowsFeature RSAT-AD-PowerShell }
$ADDomainName = $(Get-ADDomain).NetBIOSName

$gMSAFullName = "$ADDomainName\$gMSAName"
if ($gMSAFullName[-1] -ne "$") { $gMSAFullName += "$" }


# Remove the next line if you are on a DC server

if ($((Get-CimInstance -ClassName Win32_ComputerSystem).DomainRole) -in @(0..3))
```

```
if ($gMSAFullName[-1] -ne "$") { $gMSAFullName += "$" }


# Remove the next line if you are on a DC server

if ($((Get-CimInstance -ClassName Win32_ComputerSystem).DomainRole) -in @(0..3))
{
    if ($gMSAFullName -notin (Get-LocalGroupMember -Group "Administrators").Name) { Add-LocalGroupMember -(
} else {
    Write-Host "You are on a Domain Controller, make sure this user is member of domain admins" -Foregrounc
}

$ServiceCIM = Get-CIMInstance -namespace "root\cimv2" -class Win32_Service -Filter "Name='$($Service.Name)'
$ServiceCIM | Invoke-CimMethod -Name Change -Arguments @{StartName=$gMSAFullName;StartPassword=""}

Stop-Service "$($Service.DisplayName)" | Out-Null
Start-Service "$($Service.DisplayName)" | Out-Null

Give the full name of the service account (starts normally with gMSA_): : gMSA_Peppol$
```

2. Secondly, it asks for what service you want to use this.


Choose the service to change to this gMSA account

| Name | DisplayName | State | StartMode | StartName |
|---|---|---|---|---|
| AdminService11.6(64-bit) | AdminService for OpenEdge11.6 (64-bit) | Running | Auto | LocalSystem |
| AJRouter | AllJoyn Router Service | Stopped | Manual | NT AUTHORITY\LocalService |
| ALG | Application Layer Gateway Service | Stopped | Manual | NT AUTHORITY\LocalService |
| AppHostSvc | Application Host Helper Service | Running | Auto | LocalSystem |
| AppIDSvc | Application Identity | Stopped | Manual | NT Authority\LocalService |
| Appinfo | Application Information | Running | Manual | LocalSystem |
| AppMgmt | Application Management | Stopped | Manual | LocalSystem |
| AppReadiness | App Readiness | Stopped | Manual | LocalSystem |
| AppVClient | Microsoft App-V Client | Stopped | Disabled | LocalSystem |
| AppXSvc | AppX Deployment Service (AppXSVC) | Stopped | Manual | LocalSystem |
| AudioEndpointBuilder | Windows Audio Endpoint Builder | Stopped | Manual | LocalSystem |
| Audiosrv | Windows Audio | Stopped | Manual | NT AUTHORITY\LocalService |
| AutomationManagerAgent | Automation Manager Agent | Running | Auto | LocalSystem |
| AxInstSV | ActiveX Installer (AxInstSV) | Stopped | Manual | LocalSystem |
| BASupportExpressSrvcUpdater_N_Central | N-able Take Control Updater Service (N-Central) | Running | Auto | LocalSystem |
| BASupportExpressStandaloneService | SolarWinds Take Control Service | Start Pending | Auto | LocalSystem |
| BASupportExpressStandaloneService_N_Central | N-able Take Control Service (N-Central) | Running | Auto | LocalSystem |
| BFE | Base Filtering Engine | Running | Auto | NT AUTHORITY\LocalService |
| BillSync2 | BillSync2 | Running | Auto | billit@ins.w2k |

## Scheduled Tasks

Tip: You can also use gMSA's as the user for scheduled tasks (not thoroughly tested). But should work with this oneliner command prompt line (elevated)

```
schtasks /change /TN "YourTaskName" /RU "INS\gMSA_Name$" /RP ""
```