PowerShell for Beginners

Basics and Complex Exercises

Table of Contents

- Introduction
- Presentation PowerShell
- PowerShell Basics (Cmdlets, self-help)
- Pipeline, manage processes and services
- Users and groups, user profiles
- File system and NTFS-permissions, shares, network drives
- Network configuration
- Server modules, log analysis, web access, jobs
- Programming with PowerShell (ps1-scripts, accessing .NET objects)
- Create and present exams and complex exercises with PowerShell

User and Group Objects

- Requirement: CRUD pattern for user and group objects
  - Create users / groups
  - Read user / group properties
  - Update user / group behavior
  - Delete users / groups

- Appropriate Cmdlets and parameters have to be provided by PowerShell!

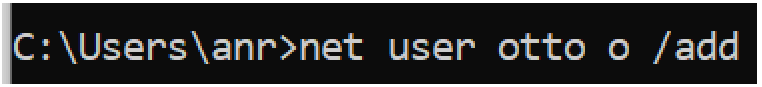
User Administration

- Local users at a workstation
  - Cmdlets of the *LocalUser* family
- Users in Active Directory (AD) environments (domains)
  - Cmdlets of the *ADUser* family
  - Only available for Windows Server 2016 upwards

Create a new User

- Cmdlet `New-LocalUser`
- ☞ The verb `Create` doesn‘t exists for Cmdlets (until today at least)

- Example:
  - User must have name and password
  - Flash back to the command prompt for user *otto* with password *o*



```
C:\Users\anr>net user otto o /add
```

👍 Easy to use, can be used in Batch scripts without modification

⚡ Password is shown in plaint text

Create a new User

- Cmdlet `New-LocalUser`
- Parameters `–Name` and `–Password`
- The Password is an object of type `System.SecureString`

- Three options to enter the parameter `–Password`:
  - Don't provide value

    → Masked password can be entered at the prompt

  - Use Cmdlet `Get-Credential` as value to the parameter

    → Enter password in pop-up window

Create a new User

- Cmdlet `New-LocalUser`
- Parameters `-Name` and `-Password`
- The password is an object of type `System.SecureString`

- Three options to enter the parameter `-Password`:
  - Directly convert value to a SecureString object

    → `(ConvertTo-SecureString -AsPlainText -Force -String “o“)`

Create a new User

- If you convert the password to a SecureString ad hoc, there is no security advantage of the `net user` command (password is plain text)

☞ Caution:

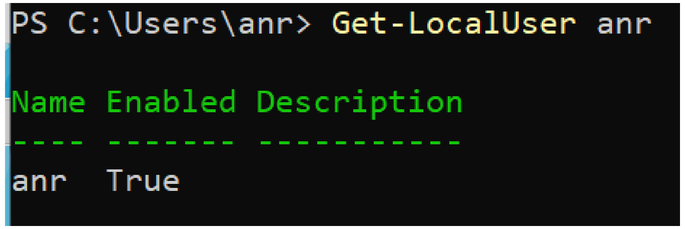
- With `net user /add` one is a member of the User group immediately.
- A `New-LocalUser` one is no member of the User group per default.
  - hence he cannot log in
  - must be added to the User group manually
  - enhances control (i.e. security)

RUD Operations for Users

- Retrieve user information with `Get-LocalUser`
- As default only the name and enabled property are show

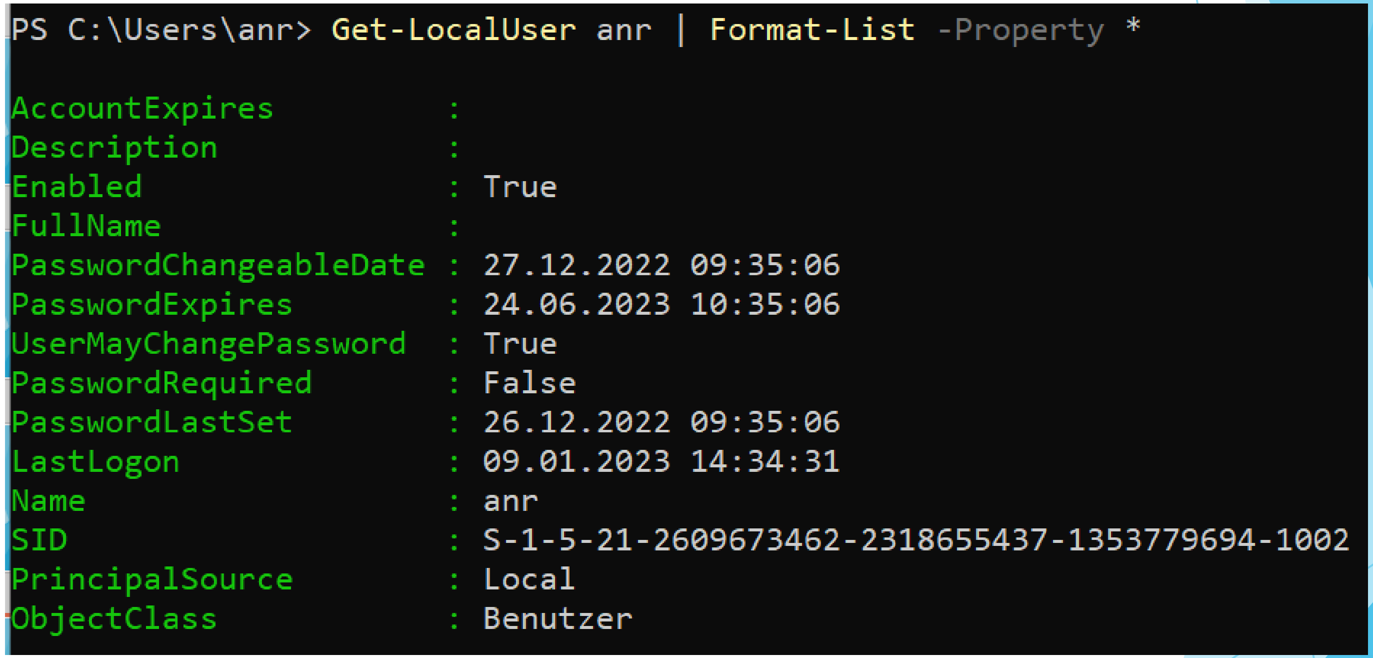
- Advice:
  - For more information format with `Format-List -Property *`
  - Alternatively use `Select-Object -ExpandProperty <PROP>`
  - Works for all other Cmdlets as well

RUD Operations for Users - Expand shown Information



```
PS C:\Users\anr> Get-LocalUser anr

Name Enabled Description
---- ------- -----------
anr  True
```



```
PS C:\Users\anr> Get-LocalUser anr | Format-List -Property *

AccountExpires         :
Description            :
Enabled                : True
FullName               :
PasswordChangeableDate : 27.12.2022 09:35:06
PasswordExpires        : 24.06.2023 10:35:06
UserMayChangePassword  : True
PasswordRequired       : False
PasswordLastSet        : 26.12.2022 09:35:06
LastLogon              : 09.01.2023 14:34:31
Name                   : anr
SID                    : S-1-5-21-2609673462-2318655437-1353779694-1002
PrincipalSource        : Local
ObjectClass            : Benutzer
```

RUD Operations for Users

- Change user properties
- Cmdlet `Set-LocalUser`

- Delete user
- Cmdlet `Remove-LocalUser`

- For details of usage and help use the already knwon helper Cmdlets

☞ Do you remember the three main helper Cmdlets ?

Managing Groups

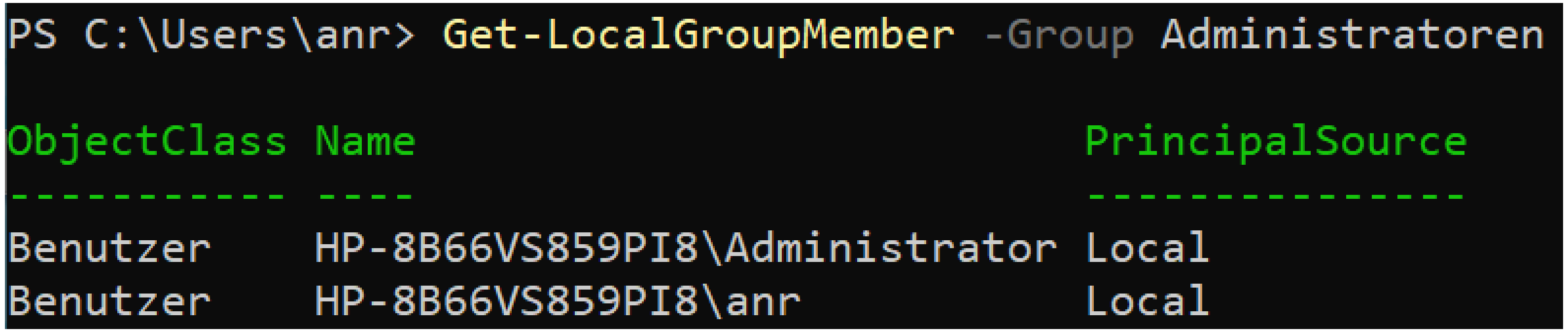
- There are several type of groups in the Windows world:
- Local groups (treated here)
- Domain local groups (Windows Server domains)
- Universal groups (Windows Server domains)
- …and then some types and subtypes, some with their own Cmdlet families

Managing Groups - CRUD-Pattern

- Cmdlets of the `LocalGroup` family for the
  - creation,
  - retrieval of properties,
  - changing / setting of properties and
  - deletion of groups

Managing Group Members

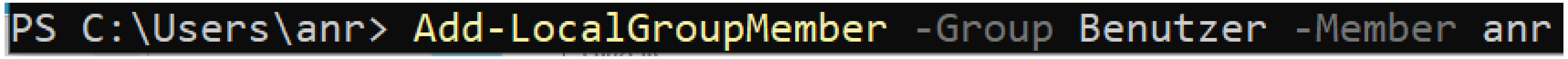
- List members of the *Administrators* group



```
PS C:\Users\anr> Get-LocalGroupMember -Group Administratoren

ObjectClass Name                              PrincipalSource
----------- ----                              ---------------
Benutzer    HP-8B66VS859PI8\Administrator Local
Benutzer    HP-8B66VS859PI8\anr           Local
```

- Add user *anr* to group *Benutzer* (engl. User) (add as member)



```
PS C:\Users\anr> Add-LocalGroupMember -Group Benutzer -Member anr
```

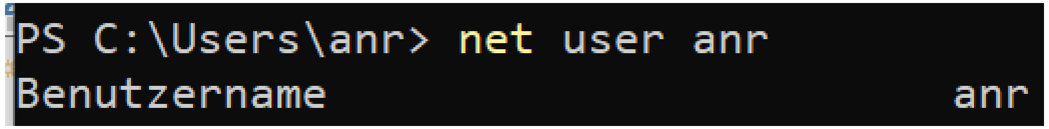
Group Membership

- Task: Show group membership of users

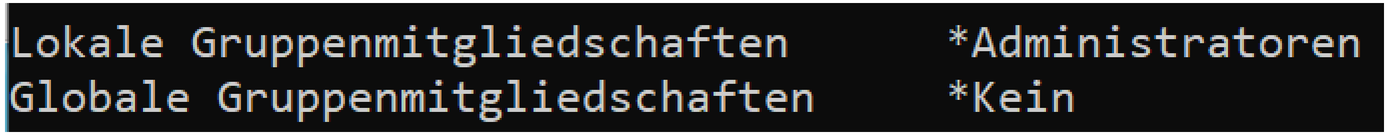
- Specialty:
- There is no (simple) possibility or Cmdlet to list all group memberships of a given user.
- Expanding properties does not lead to success

Group Membership

- Command prompt: `net user anr`
- Caution: `net user` returns a string and not a PowerShell object!



```
PS C:\Users\anr> net user anr
Benutzername                                anr
```



```
Lokale Gruppenmitgliedschaften      *Administratoren
Globale Gruppenmitgliedschaften     *Kein
```

- Best Practice:
  - Continue to use `net user` command
  - Alternative: iterate over all groups, list members and filter

User Profiles

- Profile information is stored in PS variable `$Profile`
- All in all, there are four profiles:
  - CurrentUserCurrentHost
  - CurrentUserAllHosts
  - AllUsersCurrentHost
  - AllUsersAllHosts

- Default profile is *CurrentUserCurrentHost*
- Personal settings are saved in user profile file

PowerShell Aliases

- PowerShell already contains numerous aliases
  - e.g. *gci* for *Get-ChildItem* (*cmd: dir, bash: ls*)

- Relevant Cmdlets:
  - `Get-Alias`
  - `Set-Alias`

- Purpost of aliases:
  - Abbreviation of frequently used commands
  - Comfort feature for users switching from cmd or GNU/Linux

PowerShell Aliases

- Example:
  - Clear the PowerShell screen (resp. its buffer)

Interface	Cmdlet/Command
PowerShell	Clear-Host
Windows cmd	cls
GNU/Linux bash	clear

- PowerShell implementiers `cls` and `clear` as aliases of *Clear-Host*
- Thus easier transition from *cmd* and *bash*

Exercise PS41, PS42, PS44
Users, Groups and User Profiles

- Cmdlets to create and administrate users
- Cmdlets to create and administrate groups
- Optional exercise PS43 (employing the AGP-principle)
- Create PowerShell user profiles
- Use and define PowerShell aliases