Onevinn AB

TSCommander

Installation and user's manual

CONTENTS

1. VERSION

Version	Author	Date	Remark
1.0	Johan Schrewelius	2021-04-15	Document created

2. EULA

https://www.onevinn.com/eula

3. DESCRIPTION

Onevinn TSCommander adds the possibility to run server-side commands from a running MEMCM Task sequence. It consists of two parts, an extension to the Task sequence editor in the MEMCM console and a Windows service. The service must be installed on the Site-server (PSS), the extension should be installed on any (all) computers running the MEMCM console.

TSCommander supports Cloud management gateway (CMG) or any externally published MP and adds the most common tasks, traditionally performed by scripts or webservices, as a custom Task sequence step. For example, adding or removing the computer to a Collection or AD group or moving between OUs. Custom scripts are also possible.

The solution is not new, an earlier version was available on Technet Gallery (RIP), as part of the <<SCCM Extensions>> package. It has, however, overgone a significant facelift, etw-logging has been added, Framework lifted to 4.7.2 and more.

4. REQUIREMENTS

- **DotNet Framework 4.7.2**

5. FUNCTIONALITY

When a MEMCM Task sequence runs there is sent a steady stream of status messages to the site. These messages are posted through a Management Point, processed on site server, and finally written to the site Database. By injecting a custom message in this stream and sniff the database for it we can transfer or perhaps more adequately <<order>> a certain action to be performed server side.

1. A Run server command step is executed in the TS.
2. A custom message containing the <<order>> is posted on the status message queue.
3. MEMCM transfers the message to the site DB.
4. The Onevinn TSCommand Service catch the message and performs the <<order>>.

6. AVAILABLE ACTIONS

The following <<Actions>> are available:

Action	Description	System
AddToADGroup	Adds Computer to ':' separated list of AD Groups.	AD
AddToCollection	Adds Direct Membership in ':' separated list of CollectionIDs.	MEMCM
AddToManagedBy	Adds a User to the computers ManagedBy attribute.	AD
ClearLapsPasswordTimestamp	Clear Laps Password Timestamp, forcing new password.	AD
ClearPXEFlag	Clears PXE-flag for given Computer.	MEMCM
DeleteDiscoveredSystems	Deletes Computer with provided name and a ResourceID starting with '2' (Discovered in AD by SCCM).	MEMCM
DeleteUDA	Deletes User Device Affinity.	MEMCM
DisableComputerAccount	Disables Computer Account in AD.	AD
MoveToOU	Finds and moves a Computer to provided Target OU (DN).	AD
RemoveFromADGroup	Removes Computer from ':' separated list of AD Groups.	AD
RemoveFromCollection	Removes Direct Membership from ':' separated list of CollectionIDs.	MEMCM
RemoveFromOSDCollection	Removes Direct Membership from ':' separated list of CollectionIDs. Clears PXE-flag.	MEMCM
SetComputerDescription	Sets or updates the description of a Computer in AD.	AD
ClearComputerDescription	Clears the description of a Computer in AD.	AD
Custom	Run a custom script server side. All TS-Variables with the addition of %ResourceID% can be used.	N/A

7. ACCOUNT AND PERMISSIONS

The Onevinn TSCommand service is run under a service account, it can perform action on AD object and on devices in MEMCM. It is also requiring select permissions on a View in the site Database (CM_xxx), << v_StatMsgWithInsStrings >>.

This manual will not describe how to create a custom RBAC role in MEMCM or how to delegate permissions in Active Directory.

To cover the requirements, we recommend:

MEMCM:

Role <<**Operations Administrator**>>, **all Scopes**.

Database:

Make the account a member of the local user group <<**ConfigMgr_DViewAccess**>>, this group resides on the DB server, so if you are using a remote SQL, that is where to look.

Active Directory:

This completely depends on which Actions you wish to be able to perform from the Task sequence. If you for example intend to use the action <<MoveToOU>> you will have to allow the service account to perform the move. Same rule applies to all other available AD actions, the service account must be allowed to perform the action. As always, it is recommended to keep all permissions at a minimum level.

8. INSTALLATION

The setup is divided in **two installers**.

The Windows service:

"OnevinnTSCommandExecuter <version>.msi"

The Console / TS Editor extension:

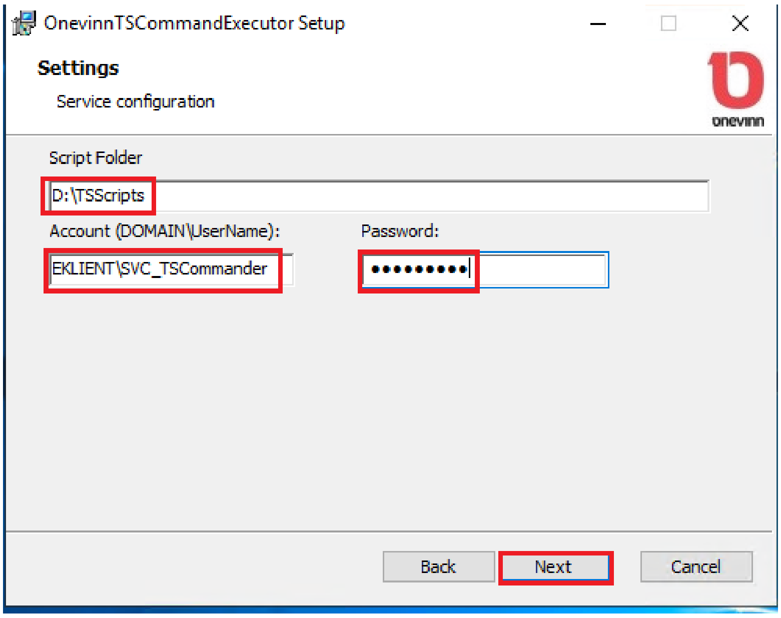
"OnevinnRunServerCmd <version>.msi"

8.1. Setup service

Before installing the service make sure you have performed the necessary role assignment and group membership explained above, otherwise the service will not be able to start and the installation fail.

1. Double click the msi, click Next, accept EULA and click Next again.

2. On the setting page fill in “Script Folder”, this is the location where the service will expect to find custom scripts called from the Task sequence, and you service account and password.



OnevinnTSCommandExecutor Setup

Settings

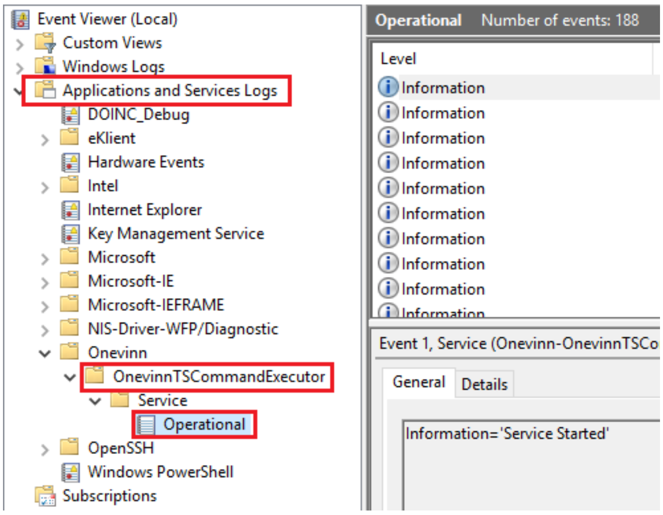
Service configuration

Script Folder: D:\TSScripts

Account (DOMAIN\UserName): EKLIENT\SVC_TSCommander

Password: •••••••••

3. Proceed by clicking Next, Next, Install and Finish.
4. Examen the Eventlog for possible errors.



Event 1, Service (Onevinn-OnevinnTSCo

Information='Service Started'

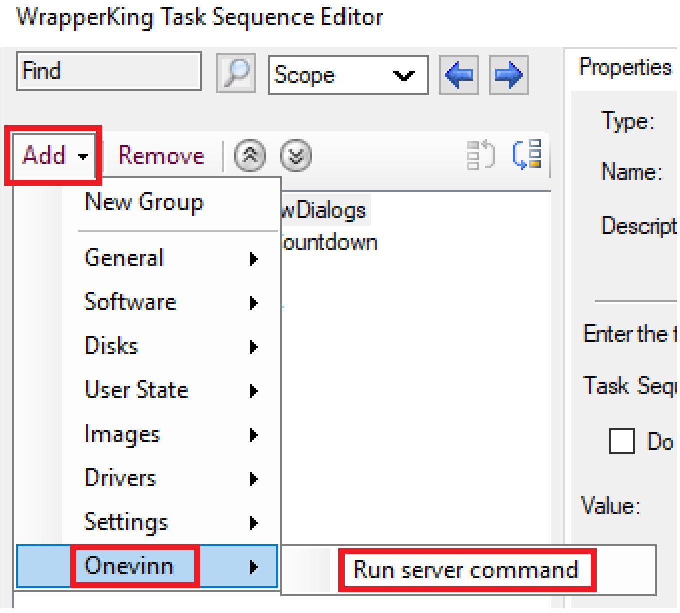
8.2. Install the extension.

The TS Editor extension will have to installed on every computer running the MEMCM console. At locations where you do not require the possibility to view or edit Task sequences it can be skipped.

Make sure not to have any running instances of the MEMCM console during the installation.

1. Double click the installer (msi) and follow the instructions on screen, no configuration is required.

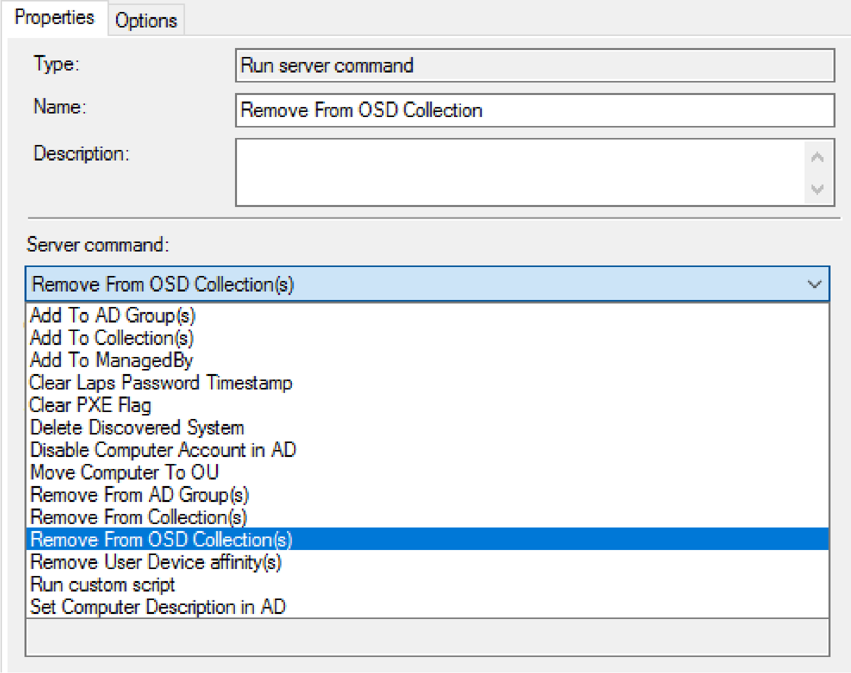
2. Start the MEMCM console and open a Task sequence for editing, a new custom step should now be available:



9. USAGE

Once a <<Run server command>> step has been added to the Task sequence chose which <<Action>> should be performed and provide the necessary argument for it.

Example #1: <<**Remove From OSD Collection**>>



Type: Run server command

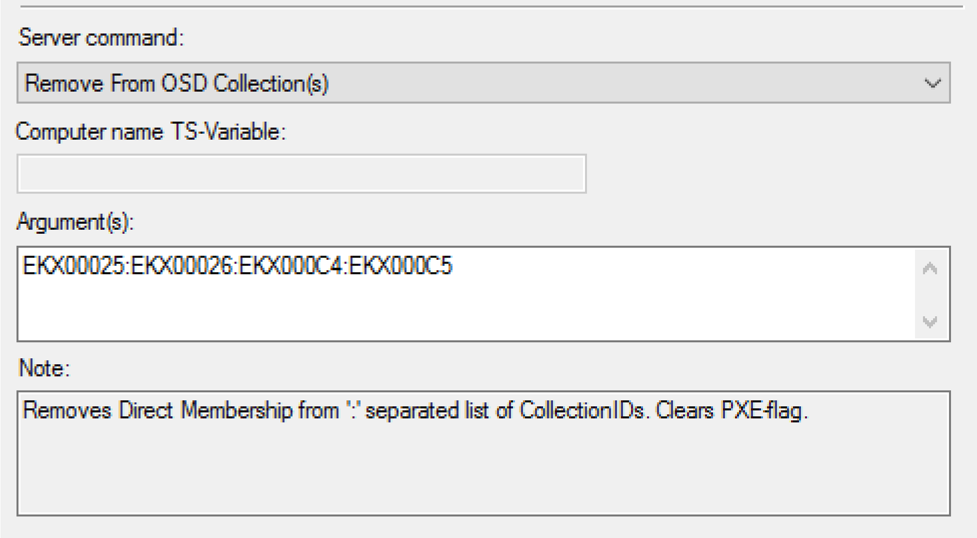
Name: Remove From OSD Collection

Description:

Server command: Remove From OSD Collection(s)

- Add To AD Group(s)
- Add To Collection(s)
- Add To ManagedBy
- Clear Laps Password Timestamp
- Clear PXE Flag
- Delete Discovered System
- Disable Computer Account in AD
- Move Computer To OU
- Remove From AD Group(s)
- Remove From Collection(s)
- Remove From OSD Collection(s)
- Remove User Device affinity(s)
- Run custom script
- Set Computer Description in AD

The “Note” box will provide basic help regarding necessary argument(s), in this case we need to fill in one or more CollectionIDs. If more than one, separate by colon.



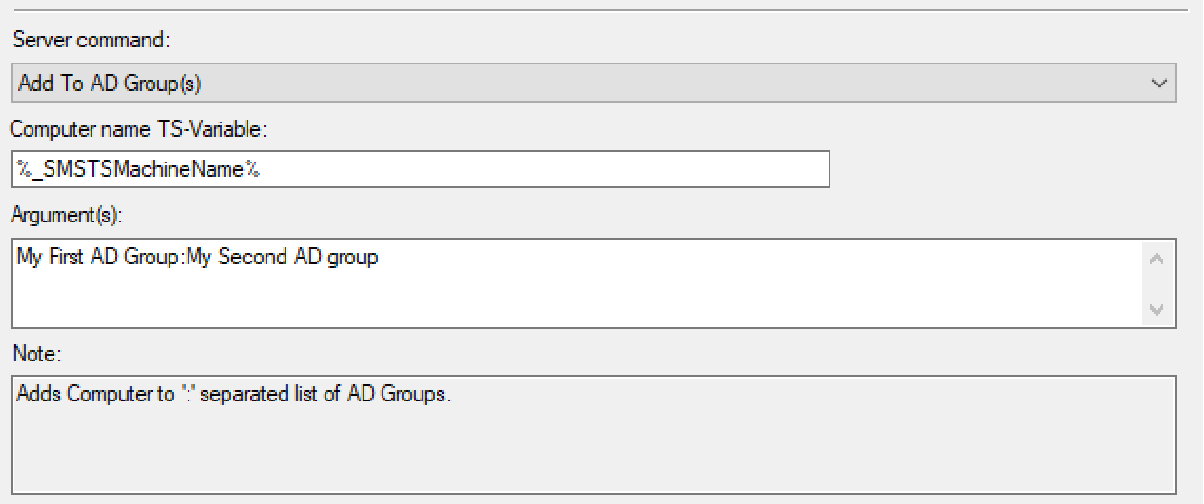
Server command: Remove From OSD Collection(s)

Computer name TS-Variable:

Argument(s): EKX00025:EKX00026:EKX000C4:EKX000C5

Note: Removes Direct Membership from ':' separated list of CollectionIDs. Clears PXE-flag.

Example #2: <<**ADD To AD Group(s)**>>



Server command:

Add To AD Group(s)

Computer name TS-Variable:

%_SMSTSMachineName%

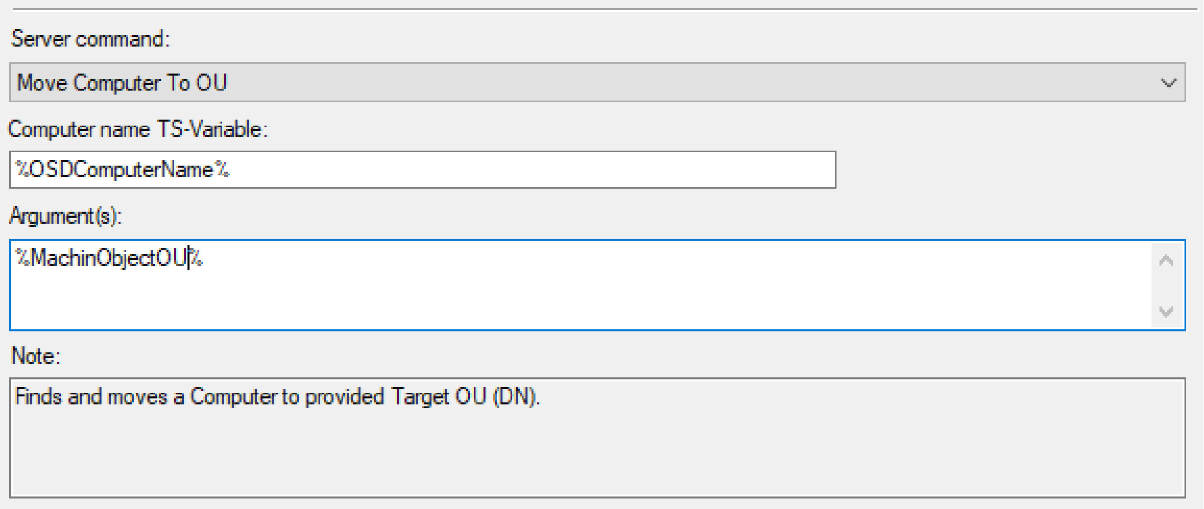
Argument(s):

My First AD Group:My Second AD group

Note:

Adds Computer to ":" separated list of AD Groups.

Example #3: <<**Move Computer To OU**>>



Server command:

Move Computer To OU

Computer name TS-Variable:

%OSDComputerName%

Argument(s):

%MachinObjectOU%

Note:

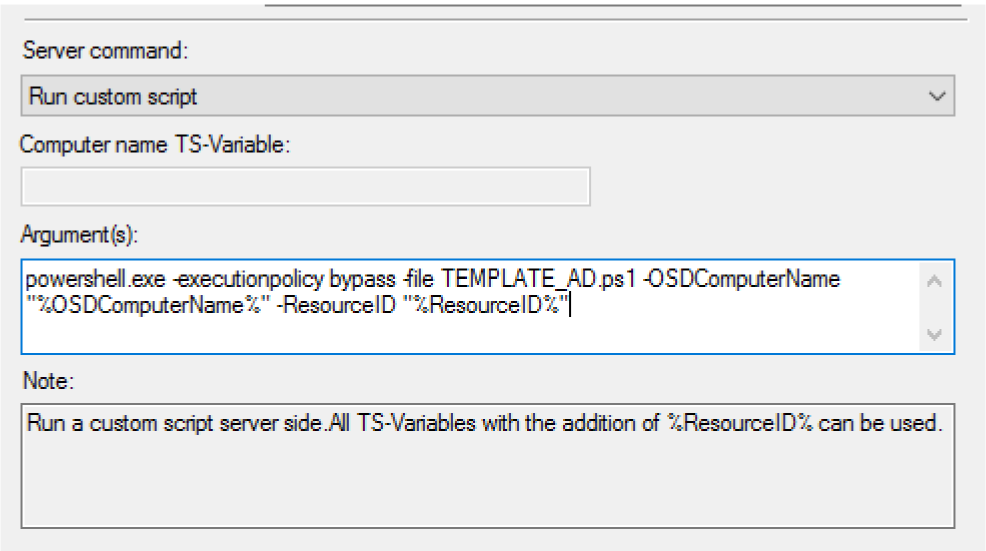
Finds and moves a Computer to provided Target OU (DN).

In this case the distinguishedName of the OU has been stored in a TS variable, one could equally well put in the actual DN: “OU=Desktops,OU=Clients,OU=eKlient,DC=eklient,DC=lab”

Example #2 and #3 uses different computer name variables to illustrate the possibilities, use what is relevant in you environment.

Example #4: <<**Run custom script**>>

In the event the built-in commands are not enough it's possible to run **custom scripts**.



Server command:

Run custom script

Computer name TS-Variable:

Argument(s):

```
powershell.exe -executionpolicy bypass -file TEMPLATE_AD.ps1 -OSDComputerName
"%OSDComputerName%" -ResourceID "%ResourceID%"
```

Note:

Run a custom script server side.All TS-Variables with the addition of %ResourceID% can be used.

In this case a script called "TEMPLATE_AD.ps1" is run with parameters -OSDComputerName and -ResourceID.

Any script in the "**Scipts Folder**" can be invoked the same way.

The service account used for TS Commander will, depending on which functionality is invoked, need matching permissions in AD, MEMCM or any other external system related to the functionality of the script.

10. ISSUES

None currently.

Keep in mind that this solution, as any other method to achieve the same functionality, is depending on network connectivity, it might a could idea not to position a <<Run server command>> step immediately after a restart computer step. You need to test your implementation and usage.