Building PoshCAT Part 1 – Create different Client Actions lists

Last week I published our new tool PoshCAT and in the upcoming weeks I will cover different things. In this blog post I will show how to create different Client Action lists for different support groups.

PoshCAT uses XML based configuration file for different *Client Actions* that you can execute through UI. One of the main ideas was that the tool should be customizable:

· Ability to add and remove commands from UI

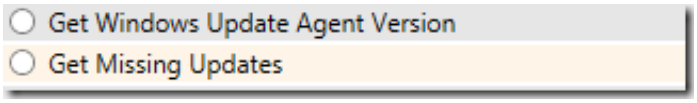
· Ability to add your own custom actions/functions

By default there are over 30 commands that are ready for use.

Commands.xml configuration

· **TASK** – Client Action name in UI

· **Component** – *PoshCAT* uses this property to group different Client Actions



· **ScriptBlock** – function name for specific task

· **Report** – this property can be *TRUE*/*FALSE*. If the value is *TRUE*, then the tool creates automatically *CSV* report. This property only works, if the function is configured correctly to return *PSObject*

· **JobType** – *PoshCAT* supports two different job types – *LOCAL* or *REMOTE*. If the *JobType* property is *LOCAL,* then it uses **Start-Job** cmdlet, for example commands like *Ping Computer*, *Restart Computer* etc. and if there is no *JobType* property, then it uses **Invoke-Command** to execute the command on remote computer.

```
<Add TASK="Get Endpoint Protection Applied Policies" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-EPAppliedPolicies" Report="True"/>
<Add TASK="Get Endpoint Protection Last Scan Date" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-EPlastScanTime" Report="True"/>
<Add TASK="Restart Computer" Component="Other Actions" ScriptBlock="Restart-CMClientComputer" Report="false" JobType="Local"/>
```

Screenshot from commands.xml

· **Parameter** – *PoshCAT* uses this property as a function parameter, for example actions like *Hardware Inventory Cycle, Set ConfigMgr Client Cache Size* etc.

```
<Add TASK="Hardware Inventory Cycle" Component="Client Schedule Actions" Parameter="00000000-0000-0000-0000-000000000001" ScriptBlock="Invoke-ClientSchedule" Report="True"/>
<Add TASK="Software Inventory Cycle" Component="Client Schedule Actions" Parameter="00000000-0000-0000-0000-000000000002" ScriptBlock="Invoke-ClientSchedule" Report="True"/>
```

Screenshot from commands.xml

How to create your own Client Actions list

Let´s assume that you want to give this tool to Desktop Management guys and you want to limit the command lists. In this example I will only keep “**Software Updates and Endpoint Protection Actions”** actions but you can use the same approach to remove other commands.

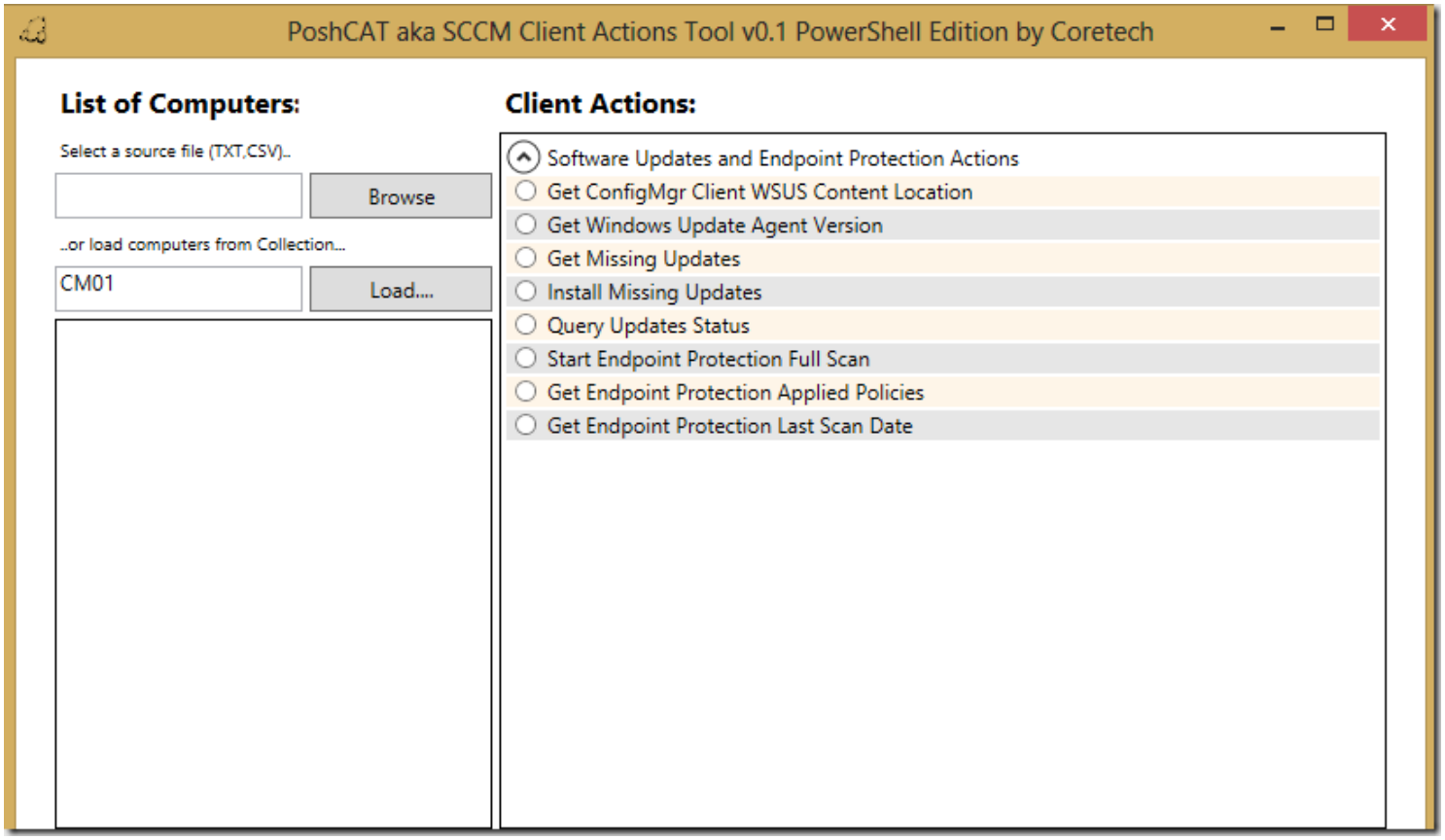
1. Create a backup file from **Commands.xml**

2. Open **Commands.xml** file and delete all the components except “**Software Updates and Endpoint Protection Actions**” and the new **Commands.xml** looks like this:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration version="1.0" encoding="utf-8">
<ToolActions>
  <Add TASK="Get ConfigMgr Client WSUS Content Location" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-CMClientWSUSContentLocation" Report="True"/>
  <Add TASK="Get Windows Update Agent Version" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-CMClientWUAVersion" Report="True"/>
  <Add TASK="Get Missing Updates" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-CMClientMissingUpdates" Report="True"/>
  <Add TASK="Install Missing Updates" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Install-CMClientMissingUpdates" Report="false"/>
  <Add TASK="Query Updates Status" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-WindowsUpdateStatus" Report="True" Parameter="UserPrompt"/>
  <Add TASK="Start Endpoint Protection Full Scan" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Start-EPFullScan" Report="false"/>
  <Add TASK="Get Endpoint Protection Applied Policies" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-EPAppliedPolicies" Report="True"/>
  <Add TASK="Get Endpoint Protection Last Scan Date" Component="Software Updates and Endpoint Protection Actions" ScriptBlock="Get-EPlastScanTime" Report="True"/>
</ToolActions>
<CMServer>
    <Name>CM01</Name>
</CMServer>
</configuration>
```

3. Save the **Commands.xml** file

4. Start **PoshCAT** and you should see the following UI



Enjoy!