# ketterhead peelbackIntroduction

The Shared Device Module is designed to switch the assigned user of a device within Workspace ONE UEM (WS1). Assignment of a device to a user is important for Service Desk / Inventory functions and also allows the Workspace ONE Hub (Catalogue) app to work.

The module also moves the device to a designated Organization Group (OG) or will keep the device in the current OG based upon the desired configuration. The script uses API calls to the WS1 API server(s) to determine if the device is set as ‘Corporate – Shared’. If it is not, the script exists, and no action occurs. This means the module can be deployed widely.

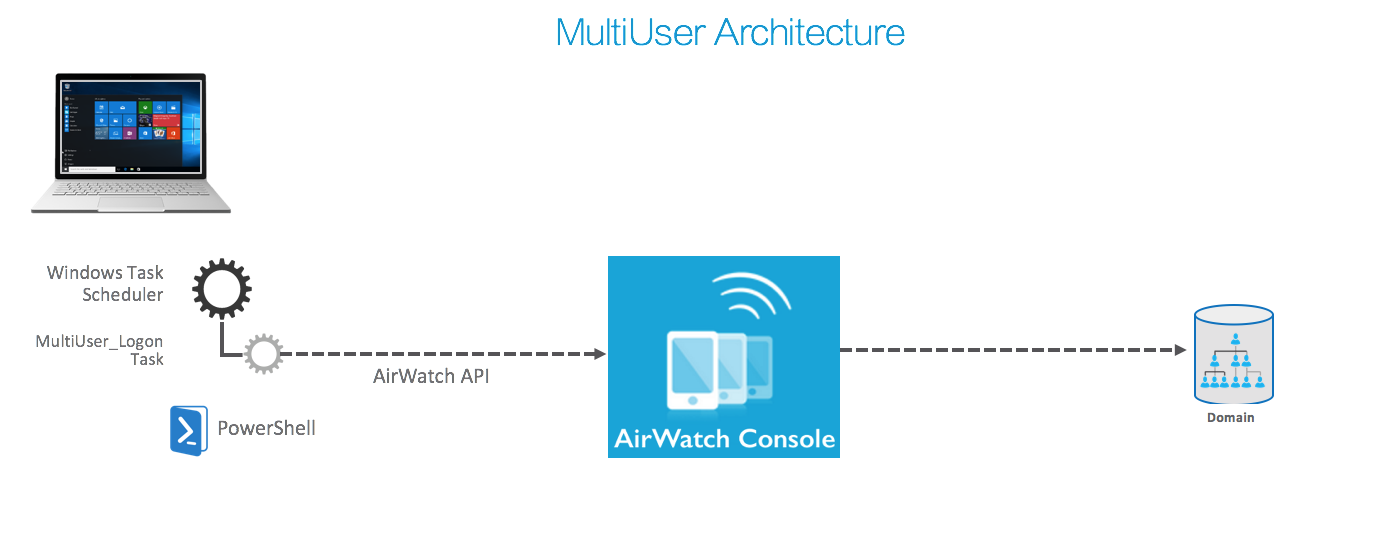
The module has also been designed with the following scenarios in mind:

* Standard User logon
* Screen locked, second user logon
* Screen unlock, user logon
* Device moved to designated OG (configurable)
* Device kept in current OG (configurable)

The module is provided as-is, with best effort support and is based upon components included in the Custom Device Inventory found on <https://code.vmware.com>.

# How It Works

The Shared Device Module for Windows 10 is relatively simple – outlined below are the steps and components used:



1. A Scheduled Task runs ‘At Logon’ as well as at ‘Workstation Unlock from any user’ and runs the AWLogon.ps1 script.
2. The script reads the shared.config file in the Shared folder.
3. The AWLogon.ps1 script then calls functions in various scripts in the Shared folder to determine the following:
   1. If the device is set as ‘Corporate – Shared’. If yes:
      1. Who the current logged on user (Active and Console) is
      2. Does the current logged on user match the assigned user, if no:
         1. Assign the current logged on user to the device
      3. If shared.config LogonGroup set to “Current”, leave the device in current OG, if not:
         1. Determine current OG, if the same as designated OG, leave device
         2. (to be confirmed if required) Move the device to configured LogoffGroup OG to force a refresh of the Smart Groups for the device
         3. Move the device to configured LogonGroup OG

See the CustomInventory\_Guide documentation within the [Windows - Custom Device Inventory - Smarter Groups - Group Policies module](https://code.vmware.com/samples/3676/windows---custom-device-inventory---smarter-groups---group-policies) found on <https://code.vmware.com>, for more information on adding security to the deployed utility.

# Implementing the Module

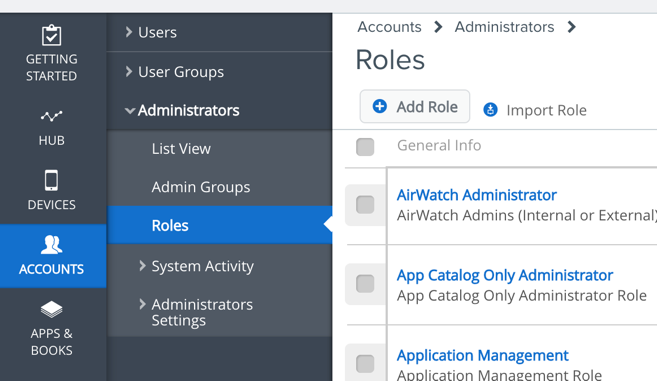
There are multiple steps and multiple files to be updated for specific environment configuration. The extracted folder should contain the following files:

A screenshot of a cell phone

Description automatically generated

## Step 1: Configure the service account for the API connection

A Basic WS1 Administrator account is used by devices for secure communication to WS1 API Servers. There are only a few admin rights needed with the primary ones being read, and only a few write permissions being required.



To simplify creating this role, you may simply import the Win10\_API\_Role.xml into WS1 found within the [Windows - Custom Device Inventory - Smarter Groups - Group Policies module](https://code.vmware.com/samples/3676/windows---custom-device-inventory---smarter-groups---group-policies) at <https://code.vmware.com>. You can add the role on the Accounts > Administrators > Roles page, then selecting the **Import Role** button, and then uploading the XML file included.

The only write commands it allows are:

* Creation of a tag
* Add/remove a tag from the current device
* Change the current device OG

Next, create a new Basic WS1 Administrator account with the new Win10\_API\_Role at the top-level organization group. Record the username and password and then Base64encode them using the format:

Username:Password

<https://coderstoolbox.net/string/#!encoding=base64&action=encode&charset=utf_8> is useful for this task.

## Step 2: Configure your OGs for shared devices

Configure the shared.config file found in the UserManagement folder of the module. This configuration file contains information used to checkout/checkin the device once a user assignment has been made. It is advised to utilise OGs below the Production OG if Tier 1 or common applications are assigned to the OG layer with the intent of deploying to all devices. The action of moving OGs will force applications not assigned to that OG or inherited from a higher OG to be removed.

The following process should be followed to update the necessary attributes:

* Create the Logoff and Logon OGs within the WS1 Console  
    
  A screenshot of a cell phone

  Description automatically generated
* Edit the shared.config file and update the LogoffGroup and LogonGroup attributes with the OG Group ID  
    
  **{  
   "SharedConfig": {  
   "LogoffGroup":"sharedunassigned",  
   "LogonGroup":"sharedassigned",  
   "StagingUser":"staging@sharedunassigned.com"  
   }  
  }**
* Update the StagingUser attribute with the email address of a basic user  
  A screenshot of a cell phone

  Description automatically generated

The best practice for this solution is to have two nested OGs for the devices, so that the script can initiate an update of the smart groups the device is a member of.

It is possible to set the LogonGroup to “Current” in order to keep the device in the same OG, effectively only changing user assignment. This configuration will reduce the number of API calls, simplify deployment in deployments with complex OG hierarchies, and reduce the risk of applications being uninstalled and installed due to differing assignment at the OG layer. It is also possible to have non-nested OGs.

Note, that whilst the LogoffGroup is not utilised at present, the OG must exist.

## Step 3: Update the api.config file to match your environment

The api.config file found in the Shared folder of the module contains information to connect securely to the REST API interface of the WS1 environment. This file is read during installation, encrypted and stored in the registry of the machine to prevent credentials being compromised, then deleted during installation.

The following process should be followed to edit the api.config file and update the necessary attributes:

**{**

**"ApiConfig": {**

**"DeviceId": "",**

**"OrganizationGroupId": 570,**

**"Server": "https://APISERVER.awmdm.com",**

**"ApiKey": "rQKhu/SVXyBVfGH6K31OTFYnT97/h7PRsn6Zr7z+Zb8=",**

**"ApiAuth": "Basic d2luMTBhcGl1c3NFGkVjbjBPTzNFazM3JA==",**

**"SSLThumbprint": "26f16b9616791ad818fa33c05070c0453743b84c"**

**}**

**}**

* Update the OrganizationGroupId attribute with the number of the highest level OG. You can find this # on the Organization Groups detail page URL.  
  A screenshot of a cell phone

  Description automatically generated
* Update the server to your Server attribute with your API Server FQDN, eg:  
    
  <https://APISERVER.awmdm.com/API/help>
* Update the APIKey attribute with your API Key - All Settings 🡪 General 🡪 Advanced 🡪 API 🡪 REST. You can utilise the default AirWatchAPI key or create another.
* Update the APIAuth attribute with the Base64 encoded username:password from Step 1. Ensure the Word “Basic “ preceeds the Base64 encoded username:password.
* Update the SSLThumbprint attribute by getting the Thumbprint from the API Server SSL Certificate when connecting with a web browser.
  + Web browse to <https://APISERVER.awmdm.com/API/help>
  + Click on certificate > Details > Thumbprint
  + Copy Thumbprint and paste into SSLThumbprint, removing spaces

## Step 4: Deploy to devices

Finally, package and deploy the module to devices.

* ZIP the contents of the module with the updated configuration files being sure not to zip the folder:  
    
  A screenshot of a social media post

  Description automatically generated
* Name the ZIP file **SharedDeviceModulev275.ZIP**. This file is deleted during installation in order to remove the API and credential details from the device. If using a different name, update the **setup.manifest** file included in the ZIP and referenced during installation.
* Create a new Internal Native Application in WS1 Console – Apps & Books > Native > Add Application:
  + Upload new ZIP file
  + Set Name:  
      
    **Shared Device Module**
  + On Files tab, set uninstall command line to:  
      
    **C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -ExecutionPolicy Bypass -File ".\Uninstall.ps1"**
  + On Deployment Options tab, Set install command line to:  
      
    **C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -ExecutionPolicy Bypass -File ".\SetupEx.ps1"**
  + Set Install Context:  
      
    **Device**
  + Set Admin Privileges:  
      
    **Yes**
  + Add criteria to determine when installation is complete:  
      
    **File Exists – C:\Temp\Shared\AirWatchAPI.psm1**
  + Assign to a SmartGroup or OG