SRC CYBER SOLUTIONS LLP
CYBER RISK SOLUTIONS

AUTOMOX

OVERVIEW

Onboarding Jumpstart Guide – Worklets and Required Software

As you begin to build out your new Automox organization, understanding what is available and having access to best practices will help you to install software where you need it, and build some amazing Worklets with less gotchas on the way.

Required Software

- Using a Required Software Policy
- How to find application names and versions for Required Software Policies

Worklets

What can you use Worklets for? Well, pretty much anything you can script! You can install software, customize settings and configurations, or copy items to your devices. Being a SaaS solution, you can apply Worklets to devices that are anywhere! They just need to have an agent and internet access.

Video: How to use Automox® Worklets™

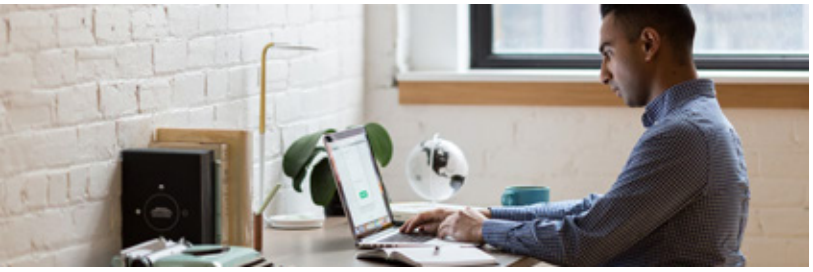
Creating a Worklet

How to Use Worklets

Evaluations

The Evaluation code block is used to determine if devices are compliant, or require remediation to bring your device into compliance. As an example, you can check if software is installed, review registry settings, check folders or files, or simply set to non-zero to continuously apply remediation scripts on a schedule.

- Evaluations and Remediations happen at different times. Evaluations run based on the scan interval set in the Group the device is placed in. Evaluations also run after policies run.



- Evaluations can be used in different ways. The way you configure the evaluation may change the status for your device. Here are a few options based on your need:
  - PRIMARY USE — define Evaluation code to verify if your Worklet is already compliant, or if it needs remediation.
  - If you would like a policy to only run on-demand and always show as compliant, you can set your Evaluation to “Exit 0.”
  - If you would like to enforce a Worklet to re-apply on a schedule, you can set the Evaluation code to “Exit -1” (or anything that is not 0). This will always show status for the devices in Groups where the policy is assigned to show as “pending.”

Remediations

- You can run policies automatically on a schedule or on-demand.
- You can run policies on-demand per device or Group.
- If you run a Worklet policy on-demand, the Evaluation is ignored, and only the remediation is run.

Considerations

Error Handling

Best Practice:

- Add the following exit code handler at the beginning of your script:

```
trap { $host .ui.WriteErrorLine( $_ .Exception); exit 90 }
```

- All scripts must return zero for success, and non-zero for failure.

NOTE: Worklets currently will not properly work as a monitoring tool. The process you run in your Worklet must exit before the agent will continue on to its next task. Please ensure your Worklet runs a process and will exit out on success or failure to keep the agent active and functional

Logging and Reporting

Logging is integral with automation. Please consider adding output to a local log, centralizing output to the Automox activity log, or both.

Worklets will return either the success or failure return stream in the activity log based on the outcome of the Worklet process. If the Worklet successfully completes, all success returns will be listed in the activity log. If the Worklet process fails, you will see the error returns in the activity log instead.

Return information from a Worklet:

• Linux — stdout or stderr to write to the activity log
• Windows — Write-Output or Write-Error to the activity log

Creating a local log — A more verbose log can be useful for audit and/or troubleshooting. Here is an example of a PowerShell installation command line including a local log creation:

```
Start-Process -FilePath 'msiexec.exe' -ArgumentList ('/i',
'7z1900-x64.msi', '/qn', "/L*V
`"C:\Windows\Temp\7z1900-x64.log`"") -Wait -Passthru
```

Windows

OS Architecture

When writing Worklets for Windows, consider the OS architecture you will be targeting. If you will be targeting both 64-bit and 32-bit devices, additional sections of code may be required to work for both.

As an example, 32-bit applications typically are installed into the "Program Files" directory on 32-bit operating systems, and in the "Program Files (x86)" directory on 64-bit systems. Another example is where SOFTWARE registry keys are stored. This can become even more complex when you consider the Automox agent is a 32-bit application, and will run 32-bit PowerShell by default.

Running Profile

Automox runs a system. Worklets will not have access to current user resources such as mapped drives, or user based network share permissions.

[System vs User]

[How to Manage User Profiles]

Testing — Best Practices

Testing your scripts locally before attempting to run them through a Worklet can save a good deal of time and help to avoid applying changes to remote devices that may be difficult to reverse.

PowerShell ISE x86 — Comes pre-installed on modern versions of Windows, and comes in both 32-bit and 64-bit versions. When testing for use with Automox, use the 32-bit version which will run in the same environmental context as a script runs through the Automox agent.

Run as System

The Automox agent runs under SYSTEM context.
When testing your scripts, you can verify your scripts have access to the local resources (drives, profiles and directories, registry hives) and environmental resources (network shares, remote applications, access to other systems, and the internet) required for your script to successfully run.

One way to test your script using SYSTEM context is by using PsExec from the SysInternals suite (PsExec - Windows Sysinternals) to run Powershell ISE (x86). This will mimic both the 32-bit PowerShell environment, and running in SYSTEM context as the Automox agent will when running your script in production.

Command to launch PowerShell ISE (x86) with PsExec:

```
PsExec.exe -s -i%windir%\syswow64\WindowsPowerShell\v1.0\
PowerShell_ISE.exe
```

Enforce Windows Registry Settings Worklet

Tip: When not testing under SYSTEM context , always use an elevated Windows PowerShell ISE (x86) instance.

Tip: When you need to access 64-bit resources like HKLM:\SOFTWARE and don't want your code to redirect to the WOW6432Node, try wrapping your code in the following script block, andrunning it from an elevated PowerShell ISE (x86). It will force the code within the scriptblock to run in the appropriate environment for your system.

Example of Scriptblock

Tips

• Ensure your Evaluation and Remediation code properly exit (even on failure). If a process is left running, it can cause the Automox agent to hang (it will wait for the process to complete before additional actions can be applied).

• Worklets are not currently built to run as monitoring processes for this reason. Monitoring processes is something we are working on for a future feature improvement.

• Consider adding output logging into your Remediation code. It will be written into your activity log and made available in the Activity Log Report.

• Content uploaded for Required Applications or Worklets will be stored in Amazon s3. If you use a proxy application or strict firewall rules, consider whitelisting automox-policy-files.s3.us-west-2.amazonaws.com

• For backwards compatibility, Automox uses -NoProfile. When invoking web requests in a Worklet, consider adding support for newer security protocols to ensure the connection is made. Here is an example of some code you can add to your scripts in this scenario:

```
"[Net.ServicePointManager]::SecurityProtocol
= [Net.SecurityProtocolType]::Ssl3, [Net.
SecurityProtocolType]::Tls, [Net.
SecurityProtocolType]::Tls11, [Net
SecurityProtocolType]::Tls12;"
```

• Further, you can define the required protocol only.

Examples

Windows Executable (.exe) Evaluation and Remediation

Evaluation using 64-bit script block (will work for 64-bit OS):

Example EXE Evalution Script Block

Remediation — Install .exe with switches and msi pass through switches

Example Remediation code as a function. This could be used as part of a more complex script.

Microsoft Installer (.msi) Evaluation and Remediation

Evaluation — Example (will work for 32-bit and 64-bit software):

Example .msi Installer Evaluation and Remediation — Evaluate for either 32-bit or 64-bit software via registry.

This example is based off of the following Worklet to Uninstall existing software: Worklet: Enforced Application Uninstall for Windows — Worklets

Example Code

Remediation — Install .msi

Example Code

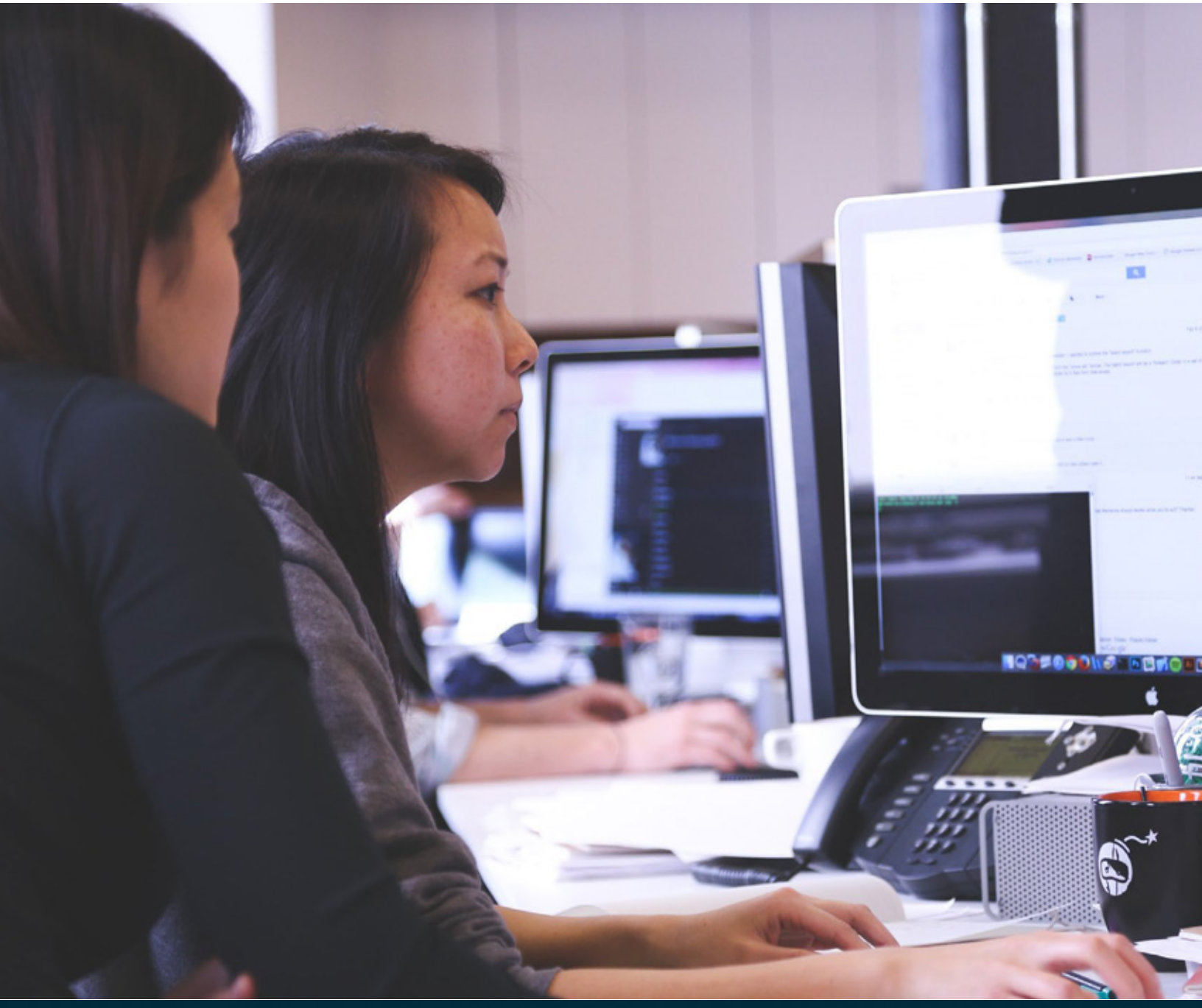
macOS

Linux

Community Resources

Community Worklets

Worklets



www.srccybersolutions.com
+91 120 232 0960 / 1
sales@srccybersolutions.com