CUSTOM TEMPLATE GUIDE

Server & Application Monitor

Version 6.8

© 2018 SolarWinds Worldwide, LLC. All rights reserved.

This document may not be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the prior written consent of SolarWinds. All right, title, and interest in and to the software, services, and documentation are and shall remain the exclusive property of SolarWinds, its affiliates, and/or its respective licensors.
SOLARWINDS DISCLAIMS ALL WARRANTIES, CONDITIONS, OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION NONINFRINGEMENT, ACCURACY, COMPLETENESS, OR USEFULNESS OF ANY INFORMATION CONTAINED HEREIN. IN NO EVENT SHALL SOLARWINDS, ITS SUPPLIERS, NOR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY, EVEN IF SOLARWINDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The SolarWinds, SolarWinds & Design, Orion, and THWACK trademarks are the exclusive property of SolarWinds Worldwide, LLC or its affiliates, are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other SolarWinds trademarks, service marks, and logos may be common law marks or are registered or pending registration. All other trademarks mentioned herein are used for identification purposes only and are trademarks of (and may be registered trademarks) of their respective companies.
# Table of Contents

- **Create custom templates in SAM** ................................................................. 5  
  - Understanding custom templates ................................................................. 5  
  - Import and export templates ........................................................................ 6  
  - Examples of custom templates ....................................................................... 6  
- **Best practices for SAM templates** ............................................................... 7  
  - Performance enhancements ............................................................................. 7  
  - Script, monitor, and template testing ............................................................ 7  
  - Script best practices ....................................................................................... 8  
- **Create a template in SAM** ........................................................................... 11  
  - Create a template ............................................................................................ 12  
  - Customize the new template .......................................................................... 13  
  - Add component monitors ............................................................................... 14  
  - Assign a template to a node .......................................................................... 15  
- **Import and export templates in SAM** ......................................................... 16  
  - Export a template ............................................................................................ 16  
  - Import a template ............................................................................................ 17  
- **Create Scripts for SAM Script Monitors** .................................................. 19  
  - Writing scripts ................................................................................................. 19  
  - Returned metrics and output .......................................................................... 20  
  - Script example ............................................................................................... 21  
- **PowerShell scripting in SAM custom templates** ........................................ 24  
  - Elements of a PowerShell script ..................................................................... 24  
  - Field descriptions ........................................................................................... 24  
  - Display SAM templates that support PowerShell scripts ............................... 26  
  - Create a Windows PowerShell Script monitor ............................................. 26  
- **Nagios scripting in SAM custom templates** ................................................. 29  
  - Nagios support in SAM ................................................................................. 29  
  - How the Nagios script works .......................................................................... 29
Components of a Nagios script ................................................................. 30
Use Nagios plugins and compiled binary files ......................................... 31
Create a Nagios Script monitor ................................................................. 31
Add a Nagios monitor ............................................................................. 32

**Linux/Unix scripting in SAM custom templates** ......................................... 34
Linux/Unix script support in SAM .............................................................. 34
How the Linux/Unix script works .............................................................. 34
Components of a Linux/Unix script ........................................................... 35
Create a Linux/Unix monitor ..................................................................... 36

**Windows scripting in SAM custom templates** ........................................ 38
Windows script support in SAM ................................................................. 38
How the Windows script works ................................................................. 38
Components of a Windows Script .............................................................. 39
Create a Windows Script monitor ............................................................ 40
  Adapt an existing Visual Basic script to a Windows Script Monitor in a new template .................. 42
  Macros for Script Arguments ................................................................. 42
  Script exit codes and text output ............................................................ 42
  Example scripts .................................................................................. 43
Create custom templates in SAM

Application monitor templates provide a powerful, customizable method to monitor applications in your environment. They help determine where errors occur: in the network or with the application. Every template includes component monitors that return metrics for monitoring performance, troubleshooting, threshold monitoring, and application health.

SAM includes a set of out-of-the-box templates you can immediately assign to application server nodes. For customized monitoring, create new templates based on existing templates or build templates from scratch.

This guide includes best practices for templates, steps to create a template, plus how to import and export templates to and from the SolarWinds online IT community, THWACK.

Review the following sections to learn about creating scripts for SAM Script Monitors in various ways:

- PowerShell scripting in SAM custom templates
- Nagios scripting in SAM custom templates
- Linux/Unix scripting in SAM custom templates
- Windows scripting in SAM custom templates

Understanding custom templates

Watch and learn more about templates.

A template is a collection of application and component monitors, configuration settings, and scripts that collect and report data for assigned applications and nodes.

The benefit of templates is the inheritance relationship that you can use to control monitoring configurations at different levels. You can assign specific application monitors to nodes for customized monitoring for only that application. Or you can use a template to assign a set of application monitors across multiple nodes.

Note the following details about editing templates:

- If you make a change at the template level, the applications based on that template are affected.
- If you make a change at the assigned application monitor level, only the individual application is affected.

Customize templates in SAM in one of the following ways:

- Extend out-of-the-box templates with new component monitors.
- Create new templates with customized settings and scripts.
- Import templates created by other SAM customers in THWACK.

The following image details the association of templates and component monitors assigned to monitored nodes, reporting data to SAM accessible through the Orion Web Console.
Import and export templates

SAM supports the importing and exporting of templates to and from THWACK, enabling customers and SolarWinds staff to share templates that provide an array of monitoring options. After importing a SAM template into the Orion Web Console, you can further extend and configure component monitors to enhance monitoring features for the template.

Examples of custom templates

The following links provide a few excellent custom templates:

- SolarWinds MIB Database
- PowerShell SNIPPET
- APC UPS monitoring, monitors without using universal pollers
Best practices for SAM templates

SAM template creation includes more than just adding and configuring component monitors. Use these best practices, tips, and tricks about performance enhancements, testing, and scripting to create, customize, and monitor component monitors and templates.

Performance enhancements

Modify the polling frequency for performance

Depending on the length of calls and amount of data pulled for a monitor, you may want to modify the frequency. For script monitors you may need to only run the script once per day or once per week. For example, to compare MIBs using the SolarWinds MIB Database template, you may only need to run the comparison once a day or a week.

Extend the polling timeout for long calls

For scripts with lengthy calls for large amounts of data, extend the polling timeout. The default 300 seconds may not be long enough for script processing to complete. If the call may take more time, especially during peak times, increase the timeout to give the system time to complete the call. For example, for MIB database comparison scripts using the SolarWinds MIB Database template, multiple files are called, downloaded, and compared to return status messages and complete specific actions.

Enhance latency and performance by pulling multiple metrics per template

When executing script component monitors in a template, SAM affects performance and latency making calls to a target server. Complete calls for up to 10 metrics per script to reduce the number of calls, increasing performance. Depending on the size and processing of scripts, balance scripts and lengthy calls across multiple instances of a script monitor.

Script, monitor, and template testing

Check credentials and server permissions for scripts

Verify you have the correct credentials with assigned account permissions to execute scripts on the Orion Web Console and target server. Issues with scripts tend to be with credentials. The script monitors may provide fields for credentials, or you may need to provide credentials in the script code, arguments, or command line. Test the script in SAM prior to verify credentials and access.

Test scripts before monitoring

When adding and configuring script component monitors, you need to test the script. When the test completes, SAM registers each returned metric as a numbered output in the Orion SQL Database. You can configure the display of collected metrics and values through the component monitor. Each script monitor supports up to 10 different outputs.
Receive accurate node status

Until tested, scripts and component monitors return an initial unknown status. After testing, polling returns accurate application status.

Script best practices

Use code comments

Code comments help document the intent for code, decisions made, and to track changes. SolarWinds recommends using code comments to keep detailed steps and responses in your code. If additional administrators need to work in the script monitors, the comments provide context for the code.

```bash
# for a comment per line.

<#
For lengthy comments per code section.
#>
```

Do not use positional parametrization

In the command line for executing scripts, always add the parameter per value. Do not assume the position of data in the command dictates the parameter. For example, use -h for hostname.

Use a header for writing multiple scripts

Create a header in your code to reuse throughout your scripts. The header could include example code and code comments for:

- A listed of exit codes
- Set variables for return metrics commonly used in your scripts
- Use code to determine if you are testing code on the target server or the Orion system

For example, the following PowerShell code returns a message identifying if the server is a test system or the Orion server:
Additionally, you could add a step to save the code if not on the Orion server.

**Use SolarWinds macros**

When using SolarWinds macros, consider assigning them to named variables in your scripts.

The following SolarWinds macros are available for Linux/Unix, Nagios, Windows Script, and PowerShell script monitors:

- `${USER}`
- `${PASSWORD}`
- `${PORT}`
- `${Node.SysName}`
- `${Node.Caption}`
- `${Node.DNS}` - Use this instead of `${IP}`.
- `${Node.ID}`
- `${Component.ID}`
- `${Component.Name}`
- `${Application.Id}`
- `${Application.Name}`
- `${Application.TemplateId}`
- `${Threshold.Warning}`
- `${Threshold.Critical}`

**Node Custom Property Macros** `${Node.CustomPropertyName}`

**Application Custom Property Macros** `${Application.CustomPropertyName}`

For agent monitored nodes, use the macros `${Node.SysName}` and `${Node.DNS}`. The `${IP}` may return a loopback IP before polling starts.

**Report status through exit codes**

Scripts must report their status by exiting with the appropriate exit code. The exit code is used to report the status of the monitor, which is seen by the user through the interface
A script should return an exit code which results in an Up (0), Warning (2), or Critical (3) status. When one of these exit codes is received the appropriate dynamic evidence table structure is created and all further exit codes are handled correctly. If the component only returns Down (1) or Unknown (4) on first use, the appropriate dynamic evidence table structure is not created appropriately.

You must test the component monitor after entering the script to properly calibrate the monitor, generate tables, and verify correct communication between the target node, SAM, and the template.

- 0 - Up
- 1 - Down
- 2 - Warning
- 3 - Critical
- Any other value - Unknown, for example 4

Multiple options for returning exit code and message

You can return one of multiple options for exit codes and messages using IF/ELSE or case statements for your scripts.

Use error trapping to capture issues

Using error trapping code such as try/catch blocks help capture and report errors. These blocks provide better reporting of an error with detailed information for the issue.
Create a template in SAM

Watch and learn more about templates.

Templates include a number of component monitors you can add and then customize. When complete, assign the template to managed nodes to collect specific metrics at each polling.

SAM includes two options for creating new templates:

- (Recommended) Create component monitors using the Component Monitor Template Wizard and add them to a new template. This option adds instances of a component monitor type with fewer steps to create a template.
- Create a fresh template without component monitors. You will need to add component monitors manually.

Using **Script component monitors** requires additional steps and considerations, as described in:

- PowerShell scripting in SAM custom templates
- Nagios scripting in SAM custom templates
- Linux/Unix scripting in SAM custom templates
- Windows scripting in SAM custom templates

Component monitors may have prerequisites, configuration, and credentials requirements for target systems. See **Component Monitor Types**.

### Create a template

Create a template:

- Component Monitor Wizard: adds multiple instances of a selected component monitor to quickly create a template. You can customize each monitor as needed and assign the monitors to a new template all through the wizard.
- New, empty template: gives you a template to add component monitors from a list of all available monitors.

### Customize the new template

Add any additional component monitors to the new template for specific services, processes, and applications you want to monitor. Customize configurations, add scripts, and

### Add component monitors

With the base component monitors added to a new template, add additional component monitors as needed per application.

### Assign a template to a node

Templates assigned to nodes begin collecting data with the next poll, providing specific metrics and data responses based on the component monitors and configurations. For script monitors, the script collects data and returns metrics and values according to the code you enter.
Create a template

The fastest method for creating a template with a set of component monitors is using the Component Monitor Wizard. The wizard provides the most common and popular component monitors per platform. Using this wizard, you can create multiple instances of a selected component monitor to add to a new or existing template.

When done, you can edit the template to add additional component monitors as needed.

For example, you may need to monitor 20 services on servers and run custom PowerShell scripts. Creating a template using the wizard to easily generate 20 Process Monitors or Service Monitors (depending on your use case) for the services. After created, you can add additional PowerShell script monitors as needed.

- You may need to log in with an administrator account to perform this action.

- When you enter credentials for component monitors, a connection test runs. Depending on the components, you may need to enter additional server and credentials data.

1. In the Orion Web Console, click > Settings > All Settings > SAM Settings > Component Monitor Wizard.
2. Select a component monitor type to add to your application or template. These include monitors based on application category: Windows Systems, Linux - Unix Systems, VMWare Systems, and Applications.

3. Select a Target server by browsing to supported systems in your environment. You may also need to select a platform type and enter credentials for the target system.

   Select and enter all required information.

   A list of available processes, services, and performance counters displays for the server. Select options to create process monitors per service.

4. Customize the component monitor and click Next.

   If you selected a script monitor, review the following information for writing scripts:
   - PowerShell scripting in SAM custom templates
   - Nagios scripting in SAM custom templates
   - Linux/Unix scripting in SAM custom templates
   - Windows scripting in SAM custom templates

5. Create a new template using the New Application Monitor Template. Enter a name when prompted.
6. Assign the template to nodes in your environment according to server type (such as Windows or VMware).
   Expand and select one or more servers to assign the template.

7. Review the configured component monitors prepared to push to nodes. When ready, confirm the template creation.
   The template creates, populating with the component monitors and configured settings, assigned to the selected nodes.

If you do not want to use the wizard, you can also create a new, empty template and add component monitors as needed. For example, you may only want to create a template with PowerShell script monitors to gather data on your systems and applications.

1. In the Orion Web Console, click Settings > All Settings > SAM Settings and select Manage Templates.
2. Click New Template.
3. Add general information and settings for the template including the name, description, and tags.

   Make sure the name of the template does not begin or end with a space. If so, application resources may not list all applications. Edit the template name to remove any extra spaces in the name.

4. Depending on the load of the calls for the component monitors, set the polling frequency and timeout.

Continue by adding component monitors.

**Customize the new template**

With the template created, you can edit each component monitor, add additional monitors, and complete any configuration settings.

1. In the Orion Web Console, click Settings > All Settings > SAM Settings and select Manage Templates.
2. Locate and select the new template in the list using search. Click Edit.
3. Add general information and settings for the template including the name, description, and tags.
4. Depending on the load of the calls for the component monitors, set the polling frequency and timeout.
5. When done, click Submit to save the changes.

   Make sure the name of the template does not end or begin with a space. If so, application resources may not list all applications. Edit the template name to remove any ending spaces in the name.
Add component monitors

When editing a template, you can add one or more component monitors.

1. You have two options for adding component monitors. This scenario uses the Manually Add Component Monitor option.

- Manually Add Component Monitors allows you to select and add multiple types of component monitors to the template
- Browse for Component Monitors opens the Component Monitor Wizard to add multiple instances of a selected component monitor

For details on each monitor, see Component Monitor Types.

2. When added, you can modify the configurations and custom settings per monitor.

3. Click the checkbox for a component monitor, click in the Quantity field to add multiples, and click Add.

   A number of each selected component monitor add to the template with an expander.

4. Expand each component monitor to configure settings, add scripts, and more.

5. When completed, click Submit.

   As you work, you can click Save and Continue Working to save as you add and complete component monitors.
Assign a template to a node

To begin monitoring with templates, you need to assign the application templates and monitors to a node. When assigned and enabled, the template collects and reports on polling data to the node according to the application monitors and configuration settings. To add nodes, see Discover and add network devices.

To assign templates to nodes:

1. In the Orion Web Console, click Settings > All Settings > SAM Settings and click Manage Templates.
2. Check the template(s) to assign, and click Assign to Node.
3. Specify the node(s) to monitor and click Next.
4. Enter or select the appropriate credentials.
5. Click Assign Application Monitors.
Import and export templates in SAM

SAM integrates with the SolarWinds online IT community site, THWACK, so SAM customers can share templates. You can export custom templates created for applications and custom scripts to THWACK to help other customers. Exporting templates creates an XML file with an .apm-template extension and adds the template to a repository accessed by SAM.

SAM automatically connects with THWACK to list all available templates for import. You need a THWACK account to download and install templates into SAM.

Please note, content posted by other community members in THWACK is not part of the SolarWinds software or documentation that you purchased from SolarWinds. Your organization should internally review and assess to what extent, if any, such custom scripts or recommendations will be incorporated into your environment. You elect to use third party content at your own risk, and you will be solely responsible for the incorporation of the same, if any.

Export a template

- Templates can take about 5 minutes to export to THWACK. If exporting multiple templates, allow time for SAM to complete the export.

1. In the Orion Web Console, click Settings > All Settings > SAM Settings, and click Manage Templates.
2. Browse or select for a template in the list.
3. Click Import Export > Export to THWACK.
4. Enter your THWACK account credentials when prompted.
   If you need an account, click Create Account and follow the steps.
A message displays with the template exporting to THWACK. The template packages into an XML file with an extension .apm-template.

**Import a template**

1. In the Orion Web Console, click Settings > All Settings > SAM Settings, and click Manage Templates.
2. Click the Shared Templates on THWACK tab.
3. Search or browse the templates. Select a template file and click Import.

To view details about a template, click the link for the content exchange area on THWACK.
4. Enter your THWACK account credentials when prompted.  
   If you need an account, click Create Account and follow the steps.

5. The template imports into the Orion Web Console. In the dialog box, click View Imported Templates.  
   A list displays of all imported templates.

   1. If you import a template with the same name as an existing template, the name of the imported template is modified by appending \( n \) to the name, where \( n \) is an integer.
Create Scripts for SAM Script Monitors

Script component monitors in SAM give you limitless options for monitoring and returning metrics for targeted servers. Each monitor has different options to execute the script, enter credentials, set working directories, and generate and display returned metrics as output.

SAM includes the following script monitors:

- Microsoft PowerShell
- Nagios including plugins
- Linux/Unix
- Windows Script

For details on creating scripts, see the following:

- Create a Windows PowerShell Script monitor
- Create a Nagios Script monitor
- Create a Linux/Unix monitor
- Create a Windows Script monitor

For additional examples, see Manage application monitor templates and component monitors.

Writing scripts

Every script monitor accepts code through the Script Body field. Develop code against the target system directly in the SAM script monitor or using your preferred scripting environment and programming language. For example, Microsoft provides a Windows PowerShell Integrated Scripting Environment (ISE). Programming environments and applications give you script libraries and color-coded source code, sometimes easier to code with than the SAM script window.

Write or copy and paste script code into the Script Body field. Each script monitor includes additional fields and options that may be required to run the scripts. For example, PowerShell includes an arguments field and Nagios includes a command line field. Depending on the programming language and script monitor, you may need to enter the following:

- Credentials
- Hostname and target server information
- Arguments and variables

When ready, you must test the script through SAM to verify credentials and target server access and to generate the outputs.
Returned metrics and output

SAM does not support ${item} or other responses. SAM expects 1 to a maximum of 10 metrics returned with a string of Statistic and a numerical value and an optional Message. When returning multiple metrics, you can add descriptive label to the returned statistics, for example Statistic.FileSize, Statistic.MailboxCount, and Statistic.Name.

If using labels, make sure not to use spaces.

SAM supports multiple values returned by a script using the following format.

Statistic.Name1: xMessage.Name1: abc
Statistic.Name2: yMessage.Name2: abc

<table>
<thead>
<tr>
<th>DETAIL</th>
<th>REQUIRED</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistic</td>
<td>Yes</td>
<td>A numeric value used to determine how the monitor compares to its set thresholds. This must be an integer value, (negative numbers are supported). Statistic.Name1: 123 Statistic.Name2: 456</td>
</tr>
<tr>
<td>Message</td>
<td>No</td>
<td>An error or information message to be displayed in the monitor status details. Multi-line messages are supported. To use this functionality, print each line using a separate command. For example: Message.Name1: abc Message.Name2: def</td>
</tr>
</tbody>
</table>

Use a write command according to the used programming language to report back data to SAM, such as:

- For PowerShell scripts, use the Write-Host command.
- For Python, use the Print command.
- For Bash/Shell, use the Echo command.

Each returned metric generates an output. You can customize the display name and thresholds as needed. If you delete the output, the metric does not display. SAM still pulls and saves the metric to the Orion SQL database.

For best system performance, create fewer calls by pulling multiple metrics per script. Depending on the load of the calls, you may need to divide the scripts and returned metrics across multiple script monitors.
**Script example**

The [SolarWinds MIB Database](https://www.solarwinds.com/) is a custom template using the [Microsoft PowerShell script monitor](https://docs.microsoft.com/en-us/powershell/scripting/). The script in this template calculates the age difference for a new database running on the Orion server by pulling the database and comparing creation dates.

**Recommended configurations:**

- Polling frequency at 24 hours (86400 seconds) or higher
- Polling timeout set depending on your Internet Connection speed

> For reference, the MIB database file download is over 200 MB.

- Execution mode: Local Host

> Running the Web Client code over remote PowerShell may cause issues.

The script code used in this PowerShell monitor include extensive comments to clearly define the steps taken. Overall, the script:

- Sets a temporary location and names for the MIB database
- Downloads and extracts data from the zip
- Compares the creation date between the original and downloaded MIB database
- Provides content back to SAM with a message and statistic data of the amount of days older than the original
- Saves the .cfg file if you want to update it at a later time

The script used in the template is as follows:

```powershell
$MibUrl = "http://downloads.solarwinds.com/solarwinds/Release/MIB-Database/MIBs.zip"
$LocalMib = "$($env:ProgramData)\SolarWinds\Mibs.cfg"

# Temporary Location for working with the MIB
$MIBZip = "$env:temp\MIBS.zip"

# Create a the name for the MIB.cfg file
$MIBCfg = $MIBZip.Replace("zip", "cfg")

# Cleanup Temp Folder
Remove-Item -Path $MIBCfg -Force -ErrorAction SilentlyContinue

# Create new Web Client

# Download from $MibUrl to $MIBZip
$WebClient.DownloadFile($MibUrl, $MIBZip)
```
# Create Shell Object
$Shell = New-Object -com Shell.Application
$Location = $Shell.Namespace( $env:temp )

# Extract MIBs.cfg file from Zip File
$ZipFolder = $Shell.Namespace( $MIBZip )
$MibFile = $ZipFolder.Items() | Where-Object { $_.Name -eq "MIBs.cfg" }
$Location.CopyHere( $MibFile )

# Compare UTC Creation date on the newly downloaded and the original MIB.cfg
$DateNewFile = (Get-Item -Path $MIBCfg ).CreationTimeUtc
$DateOldFile = (Get-Item -Path $LocalMib).CreationTimeUtc
$DaysOld = [int]( ( $DateNewFile - $DateOldFile ).TotalDays )

Write-Host "Message: MIB Database on $( $env:ComputerName ) is $( ( $DateNewFile - $DateOldFile ).TotalDays.ToString("0") ) days old"
Write-Host "Statistic: $( ( $DateNewFile - $DateOldFile ).TotalDays.ToString ("0") )"
$ExitCode = 0

#Cleanup
Remove-Item -Path $MIBZip -ErrorAction SilentlyContinue

#We Leave the MIBs.cfg file in the $temp folder in case you want to update it.
exit $exitcode

To import the template via the Orion Web Console:

1. Click Settings > All Settings > SAM Settings, and click Manage Templates.
2. Click the Shared Templates on THWACK tab.
   
   A page opens and populates with available templates from THWACK.
3. Enter SolarWinds MIB Database in the search field.
4. Select a template file and click Import.
5. Enter your THWACK account credentials when prompted.
   
   If you need an account, click Create Account and follow the steps.
6. Click View Imported Templates to display with the imported template.

Options to extend the code:

- Add error trapping. For example, you could check and ignore or respond with an error for negative values.

- Add proxy and credentials code if your server is behind a proxy:

  ```
  # Create a Web Proxy
  $WebProxy.Address = "http://proxy.address.local:80"

  # Create Credentials for Proxy
  $ProxyCreds = New-Object -TypeName System.Net.NetworkCredential
  $ProxyCreds.UserName = "ProxyUsername"
  $ProxyCreds.Password = "ProxyPassword"

  # Assign the Creds to the Proxy
  $WebProxy.Credentials = $ProxyCreds

  # Assign the Proxy to the Client
  $WebClient.Proxy = $WebProxy
  ```

- Add code to clear out the saved .cfg file if needed.

**Troubleshooting**

If the database does not fully update, use the [Orion Permission Checker](#) or check permissions.
PowerShell scripting in SAM custom templates

PowerShell is a Windows command-line shell and scripting language built on top of the .NET Framework that you can use to create custom scripts for use in SAM templates that support PowerShell scripts, and then assign templates to the main Orion server server or remote target nodes.

The SAM Windows PowerShell monitor supports custom PowerShell scripting and you can create multiple instances of this component monitor for each template. After running a script, the Orion Platform processes the script's exit code and text output to display results in the Orion Web Console.

⚠️ Please note that custom scripts you create or download from the SolarWinds online IT community, THWACK, are not part of the SolarWinds software purchased from SolarWinds. Your organization should internally review and assess to what extent PowerShell scripts will be incorporated into your environment. You elect to utilize custom scripts at your own risk, and you will be solely responsible for the incorporation of the same, if any.

To learn about PowerShell requirements, important security considerations, installing PowerShell on servers, and enabling remote access for PowerShell, see Use PowerShell in SAM.

Elements of a PowerShell script

SolarWinds recommends writing scripts in the Windows PowerShell Script monitor to verify correct access between the main polling engine, the Orion server, and target server(s). You can write and test PowerShell scripts directly on a target server to verify credentials and target server access.

The script in the monitor should include the following elements:

- Windows credentials for monitoring that provide access to servers
- Specific arguments to pass with the script
- Exit code used to report the status of the monitor in the Orion Web Console.
- Formatted output

Test the component monitor and script before assigning and using the component monitor within a template. After testing, verify that the script generated the correct database tables for output and/or output processes.

Field descriptions

**Description**

The default description of the monitor that you can override later, if necessary. The variable to access this field is ${UserDescription}.

**Enable Component**

Determines if the component is enabled. Disabling the component leaves it in the application in a deactivated state that does not impact SolarWinds SAM application availability or status.
**Credential for Monitoring**

Windows credential for a user account with login privileges for the Orion server and rights to access the target node to perform whatever action a script needs to do. For example, if a script does something with WMI, the credentials require WMI rights on the target node.

The basic PowerShell monitor resolves credential requests to run a script. However some PowerShell commands used in scripts require the use of the ${CREDENTIAL} variable, as described in Script Body section.

**Execution Mode**

Specify where to run the PowerShell script:

**Count Statistic as Difference**

Change the statistic to be the difference in query values between polling cycles.

**Run the script under specified account**

Select this option to enable impersonation with the component's credentials.

ℹ️ This option works only in local script execution mode.

- Local Host can run scripts only locally, that is, on the Orion server. This is the default value.
- Remote Host can execute scripts remotely (on the selected target node) using the Windows Remote Management (WinRM) system component. Configure WRM separately for use with the PowerShell monitor.
- If Remote Host is selected, the following options are available:
  - Use HTTPS Protocol - if checked, specifies that the secure HTTPS protocol should be used to send and receive WS-Management protocol requests and responses. Otherwise the HTTP protocol is used.
  - URL Prefix - specifies a URL prefix on which to accept HTTP or HTTPS requests. The default is wsman.
  - Port Number - specifies the TCP port for which this listener is created. For WinRM 1.1 and earlier, the default HTTP port is 80. For WinRM 2.0, the default HTTP port is 5985.

**Script Body**

Enter PowerShell script code to run and include the ${CREDENTIAL} variable if credentials are required, as shown in this example:

```powershell
$avg = Get-WmiObject win32_process -ComputerName '${IP}' -Credential '${CREDENTIAL}' - Where-Object {$_ .Name -eq "lsass.exe" } | Measure-Object -property ReadOperationCount -Average;
```

Some PowerShell commands (such as `Get-WmiObject` as shown in the example above) require the ${CREDENTIAL} variable. The user name from the specified Credential for Monitoring is stored automatically in the ${CREDENTIAL} variable for you by the monitor. As a result, the ${CREDENTIAL} variable should not be placed in the Script Arguments field, since it is set automatically. When the script is run by PowerShell, it prompts for a password. The monitor automatically provides the password from the specified Credential for Monitoring.
Custom Properties can be passed by using the following format: \${Node.Custom.XXX} where xxx is the name of the custom property.

Scripts must report their status by exiting with the appropriate exit code. The exit code is used to report the status of the monitor, displayed in the Orion Web Console.

**Script Arguments field**

Specify arguments to pass to the script. You may include the variable \${IP}, which is replaced by the IP address of the target node. Enter arguments in a comma-separated list.

For example:

\${Component.ID},${Node.Caption},${IP},Dell.ChangeAuditor,Coordinator

**Display SAM templates that support PowerShell scripts**

To display a list SAM templates that support PowerShell scripts:

1. Navigate to the Manage Templates page.
   a. Click Settings > All Settings > SAM Settings.
   b. Click Manage Templates.

2. In the search field, type PowerShell, or browse and click the PowerShell filter.

A list of templates that use customized instances of the Windows PowerShell Monitor displays.

![PowerShell templates](image)

PowerShell-based templates created by other customers and SolarWinds employees are posted in the SolarWinds online IT community, [THWACK](https://thwack.solarwinds.com). Please note that SolarWinds does not directly support third-party applications and/or custom scripts.

**Create a Windows PowerShell Script monitor**

You can use the Windows PowerShell script monitor to run custom PowerShell scripts in SAM that monitor Windows-based computers.

Before coding and testing your script, review the following resources:
• **Use PowerShell in SAM**
• **PowerShell scripting in SAM custom templates**
• **Best practices for SAM templates**

To create a Windows PowerShell script monitor:

1. Click Settings > All Settings > SAM Settings and click Manage Templates.
   The Manage Application Monitor Templates page opens. All available out-of-the-box and custom templates display.

2. Search to locate, select, and edit an existing template or click **Create New Template**.

3. Click Add Component Monitor(s) and choose Manually Add Component Monitors.

4. Search for Powershell. When displayed, select the Windows PowerShell Monitor and click add.

   ![Windows PowerShell Monitor](image)

5. Enter all arguments and parameters for the Script Arguments.

6. Click Edit Script to enter and test the script.

7. Test the script by selecting settings for the output, specified node, and specified credentials. Click Get Script Output.
   Output results display using the script code with success or failure and any additional notes.

8. Click Get Script Output to generate the output and click Save.
   The metrics save to outputs to configure as part of the component monitor. These outputs display at polling intervals for the template.

9. When complete, click Submit to save the template.
When tested, output generates into customizable sections. Each output section is named by the Unique ID with a Display Name you can edit.

For each output, you can optionally:

- **Convert the value** using selected formulas.
- Configure the statistic thresholds to refine alerting for the component monitor. Set the warning and critical thresholds with specific values or baseline data.
- Select a roll-up status sort option to display the best or worst status up through the template and component monitor.
- Add notes to describe the script output.
Nagios scripting in SAM custom templates

The SAM Nagios Script monitor supports custom Nagios scripting on Linux-based systems. You can create multiple instances of this component monitor per template to run custom Nagios scripts.

This information details the format and usage of Nagios scripts for SAM, including script arguments and output formats. This information does not provide information on coding in Nagios. For resources to learn or use the Nagios programming language, see Nagios resources. SAM does support using Nagios plugins.

After configuring Nagios and reviewing the components of a script, create Nagios scripts using the component monitor per template.

Nagios support in SAM

SAM supports using Nagios scripting with Nagios installed and configured on target servers.

To locate a list of out-of-the-box Nagios script supporting templates, navigate to the Manage Templates page. Click Settings > All Settings > SAM Settings and click Manage Templates. In the search field, enter Nagios to filter the list. These templates use customized instances of the Nagios Script Monitor.

Credentials

To process scripts, you must include credentials with permissions to run scripts on the monitored Linux-based computer.

Nagios polled by Orion agent for Linux

If you are using this component monitor with Nagios and polled by Orion agent for Linux, you need to enter a script as follows. The argument would be python $(SCRIPT) with a body of:

```python
from random import randint
STAT_PAIRS_COUNT = 10

for stat_id in range(STAT_PAIRS_COUNT):
    print 'Statistic.random%d: %d\nMessage.random%d: %s' % (stat_id, randint(1,100), stat_id, 'random number: ' + str(randint(1,100)))
```

How the Nagios script works

The Nagios script monitor executes the script remotely using the target server you provide then reports the metrics:
1. Connects to the target server using SSH.
2. Executes the script using the command line value.
   The script executes on the target system, not the Orion server. You may need to enter additional parameters to satisfy the hostname and credentials needed to execute.
3. Captures metrics and returns the data to the monitor.
4. Closes the SSH connection.
5. SAM parses the text output, saves data, and reports the values using the output formats from the component monitor.

Components of a Nagios script

SolarWinds recommends writing scripts in the Nagios Script monitor to verify correct access between the main polling engine, SAM, and the target server(s). You can write and test Nagios scripts in your preferred ISE, connecting and testing against the target server to verify if the code functions. You must test the script through the Nagios Script monitor to ensure entered credentials and target server access, and to generate metrics output.

The script in the monitor should include:

- Nagios parameters to run the script in the Command Line field
- Nagios script code added in the Script Body field with credentials
- Exit codes to report status in the script
- Formatted output after running the script

You must test the component monitor and script before assigning and using the component monitor within a template as is. Testing the script generates the required database tables for metrics output and verifies output processes.

Script Working Directory field

Enter a working directory where the script processes when executed. This is a directory located on the target system. A popular directory is /tmp.

If using a compiled binary file (ELF) script, leave this field blank.

Script Body field

Use the Script Body field to enter your Nagios script code. You can write and compile the script in a language you prefer then copy and paste the code into this field. For example, customers have used Perl, Python, and compiled binary file (ELF). Depending on the language, you may need to take additional steps for the script and command line to execute.

If you used compiled Nagios plugins, leave this field empty.

If using a compiled binary file (ELF) script, enter a character in the field. It will not be used when executed.

Scripts must report their status by exiting with the appropriate exit code. The exit code is used to report the status of the monitor, displayed in the Orion Web Console.
**Command Line field**

Enter your Nagios parameters used to run the script.

The data you enter may also require additional information depending on the code compiler:

- For Perl and Python, enter the command line required to run the script including all parameters. Start the command with `perl` for Perl and `bash` for Python.
- For compiled binary file (ELF), enter the full path to the script and all parameters.

**Use Nagios plugins and compiled binary files**

SAM supports using compiled Nagios Plugins found through the [Nagios site](http://nagios.org) and compiled binary files (ELF) used on Linux-based computers.

For Nagios plugins:

- The plugin needs to be installed on the target system. SAM does not transfer the plugins to the server.
- When configuring the Nagios Script monitor, leave the script body field empty.
- For the Command Line field, enter a path location on the remote file system for the plugin location and any required parameters.
- Nagios plugins typically require the hostname. You may need to enter a parameter -H with the hostname. You can assign and use a variable to provide the hostname by using `${IP}` or `%{Node.DNS}` in the script.

For compiled binary files:

- Transfer the script to the target system.
- Leave the Script Working Directory blank.
- Enter any character in the Script Body. The field cannot be empty but is not used when executing the script.
- For the Command Line field, enter the full path to the script located on the target system and all Nagios parameters.

**Create a Nagios Script monitor**

The Nagios script monitor allows you to run custom Nagios scripts in SAM to monitor computers with Nagios installed. Before coding and testing your script, review the [Nagios information and components](http://nagios.org) and [best practices](http://nagios.org) for SAM.

SAM supports using compiled Nagios Plugins found through the [Nagios site](http://nagios.org) and compiled binary files (ELF) used on Linux-based computers.

**For Nagios plugins:**

- The plugin needs to be installed on the target system. SAM does not transfer the plugins to the server.
- When configuring the Nagios Script monitor, leave the script body field empty.
For the Command Line field, enter a path location on the remote file system for the plugin location and any required parameters.

Nagios plugins typically require the hostname. You may need to enter a parameter -H with the hostname. You can assign and use a variable to provide the hostname by using `${IP}` or `${Node.DNS}` in the script.

For compiled binary files:

- Transfer the script to the target system.
- Leave the Script Working Directory blank.
- Enter any character in the Script Body. The field cannot be empty but is not used when executing the script.
- For the Command Line field, enter the full path to the script located on the target system and all Nagios parameters.

Add a Nagios monitor

1. Click Settings > All Settings > SAM Settings and click Manage Templates.
   The Manage Application Monitor Templates page opens. All available out-of-the-box and custom templates display.
2. Search to locate, select, and edit an existing template or click Create New Template.
3. Click Add Component Monitor(s) and choose Manually Add Component Monitors.
4. Search for Nagios. When displayed, select the Nagios Script Monitor and click add.

5. Select the Authentication Type and matching credentials from the Credentials for Monitoring drop-down menu.
6. Enter a Working Directory, used for processing the script. A typical directory customers use is `/tmp`.
7. Enter a Command Line to run the script including all parameters.
   Depending on the type of script or plugin, the command line may require additional information and parameters. For example, the file path or hostname.

8. Click Edit Script to enter and test the script.
   Depending on the type of script or plugin, you may need to enter different script information. For example, a plugin does not require data in this field. The plugin needs to be installed on the target server.

9. Test the script by selecting settings for the output, specified node, and specified credentials. Click Get Script Output.
   Output results display using the script code with success or failure and any additional notes.

10. Click Get Script Output to generate the output and click Save.
    The metrics save to outputs to configure as part of the component monitor. These outputs display at polling intervals for the template.

11. When complete, click Submit to save the template.

When tested, output generates into customizable sections. Each output section is named by the Unique ID with a Display Name you can edit.

For each output, you can optionally:

- **Convert the value** using selected formulas.
- Configure the statistic thresholds to refine alerting for the component monitor. Set the warning and critical thresholds with specific values or baseline data.
- Select a roll-up status sort option to display the best or worst status up through the template and component monitor.
- Add notes to describe the script output.
Linux/Unix scripting in SAM custom templates

The SAM Linux/Unix Script monitor supports custom scripts on Linux-based systems. You can create multiple instances of this component monitor per template to run custom scripts.

This information details the format and usage of Linux or Unix scripts for SAM, including script arguments and output formats. This information does not provide information on coding in programming languages such as Perl or Python. Prior to using this monitor, review the Linux/Unix support in SAM to configure the Linux target server.

After configuring Nagios and reviewing the components of a script, create Linux/Unix scripts using the component monitor per template.

Linux/Unix script support in SAM

SAM supports using Linux/Unix scripting with Nagios installed and configured on target servers.

To locate a list of out-of-the-box Linux and Unix script supporting templates, navigate to the Manage Templates page. Click Settings > All Settings > SAM Settings and click Manage Templates. In the search field, enter Linux or Unix to filter the list. These templates use customized instances of the Linux/Unix Script Monitor.

Credentials

You need to include credentials that can run scripts on the monitored Linux-based computer.

Python installation

You need to verify your installation of python and python-xml. Depending on your Linux distribution, you may have need to install these libraries. You can verify your Python installed libraries with the following command:

```
python -c "import sys; from xml.dom.minidom import parseString; print sys.version"
```

See the Python site for documentation and install commands. The following command typically updates and installs the required Python libraries: apt-get install python

Depending on your Linux distribution, you may need to verify install and configure specific applications and services to use the Linux/Unix Script monitor. For example, to use the Linux/Unix Script monitor with Apache services (as with the Apache template), you need to configure Apache access. For full configuration details for all services, see Linux system configurations for Component Monitors.

How the Linux/Unix script works

The Linux/Unix Script Monitor executes the script remotely using the target server you provide then reports the metrics:
1. Connects to the target server using SSH.
2. Executes the script using the command line value.
   The script executes on the target system, not the Orion server. You may need to enter additional parameters to satisfy the hostname and credentials needed to execute.
   The file saves to the Script Working Directory on the remote server.
3. Captures metrics and returns the data to the monitor.
4. Closes the SSH connection.
5. SAM parses the text output, saves data, and reports the values using the output formats from the component monitor.

**Components of a Linux/Unix script**

SolarWinds recommends writing scripts in the Linux/Unix Script monitor to verify correct access between the main polling engine, SAM, and the target server(s). You can write and test scripts in your preferred ISE, connecting and testing against the target server to verify if the code functions. You must test the script through the Linux/Unix Script monitor to ensure entered credentials and target server access, and to generate metrics output.

The script in the monitor should include:

- Parameters to run the script in the Command Line field
- Script code added in the Script Body field with credentials
- Exit codes to report status in the script
- Formatted output after running the script

You must test the component monitor and script before assigning and using the component monitor within a template as is. Testing the script generates the required database tables for metrics output and verifies output processes.

**Script Working Directory field**

Enter a working directory where the script processes when executed. This is a directory located on the target system. A popular directory is /tmp.

**Script Body field**

Use the Script Body field to enter your script code. You can write and compile the script in a language you prefer then copy and paste the code into this field. For example, customers have used Python. Depending on the language, you may need to take additional steps for the script and command line to execute.

Scripts must report their status by exiting with the appropriate exit code. The exit code is used to report the status of the monitor, displayed in the Orion Web Console.
Command Line field

Enter your command line parameters used to run the script. Use the format of `<scripting language path> ${SCRIPT}` such as perl, python, or bash. `{SCRIPT}` is replaced by the actual file name of the script after it is deployed to the target node. A temporary file is created in temp directory for the script.

Create a Linux/Unix monitor

The Linux/Unix script monitor allows you to run custom scripts in SAM to monitor Linux-based computers. Before coding and testing your script, review the Linux/Unix information and components and best practices for SAM.

1. Click Settings > All Settings > SAM Settings and click Manage Templates.
   The Manage Application Monitor Templates page opens. All available out-of-the-box and custom templates display.
2. Search to locate, select, and edit an existing template or click Create New Template.
3. Click Add Component Monitor(s) and choose Manually Add Component Monitors.
4. Search for Linux/Unix. When displayed, select the Linux/Unix Script Monitor and click add.
5. Select the Authentication Type and matching credentials from the Credentials for Monitoring drop-down menu.
6. Enter a Working Directory, used for processing the script. A typical directory customers use is `/tmp`.
7. Enter a Command Line to run the script including all parameters using the format `<scripting
language path> $\{\text{SCRIPT}\}$ such as perl, python, or bash.

Depending on the type of script, the command line may require additional information and parameters. For example, the file path or hostname.

8. Click Edit Script to enter and test the script.

9. Test the script by selecting settings for the output, specified node, and specified credentials. Click Get Script Output.

Output results display using the script code with success or failure and any additional notes.

10. Click Get Script Output to generate the output and click Save.

The metrics save to outputs to configure as part of the component monitor. These outputs display at polling intervals for the template.

11. When complete, click Submit to save the template.

When tested, output generates into customizable sections. Each output section is named by the Unique ID with a Display Name you can edit.

For each output, you can optionally:

- Convert the value using selected formulas.
- Configure the statistic thresholds to refine alerting for the component monitor. Set the warning and critical thresholds with specific values or baseline data.
- Select a roll-up status sort option to display the best or worst status up through the template and component monitor.
- Add notes to describe the script output.
Windows scripting in SAM custom templates

The SAM Windows Script monitor supports custom scripts on Windows-based systems. You can create multiple instances of this component monitor per template to run custom scripts.

This information details the format and usage of Windows Scripts for SAM, including script arguments and output formats. This information does not provide information on coding in programming languages such as those listed in the Script Engine field.

After configuring Windows and reviewing the components of a script, create Windows scripts using the component monitor per template.

Windows script support in SAM

SAM supports using Windows scripting on target Windows-based computers.

To locate a list of out-of-the-box Windows Script supporting templates, navigate to the Manage Templates page. Click Settings > All Settings > SAM Settings and click Manage Templates. In the search field, enter Windows to filter the list. These templates use customized instances of the Windows Script Monitor.

Use different scripting languages

On Windows-based computers, you can use one of several scripting languages or install a non-native scripting language.

1. Install the scripting language engine or software on the Orion server.
2. When installed, verify that the install location for the scripting language is in the system environment path variable.
3. In the SAM template Windows script monitor, enter the scripting language in the Script Engine.

How the Windows script works

The Windows Script Monitor executes the script on the Orion server connecting to the target server to collect metrics:

1. Executes the script using the script arguments on the Orion server.
2. The monitor connects to the target server using entered credentials for collecting metrics.
3. SAM parses the text output, saves data, and reports the values using the output formats from the component monitor.
Components of a Windows Script

SolarWinds recommends writing scripts in the Windows Script monitor to verify correct access between the main polling engine, SAM, and the target server(s). You can write and test scripts in your preferred Integrated Scripting Environment (ISE), connecting and testing against the target server to verify if the code functions. Test the script through the Windows Script monitor to ensure entered credentials and target server access, and to generate metrics output.

The script in the monitor should include:

- Parameters to run the script in the Scripts Arguments field
- Script code added in the Script Body field with credentials
- The script engine to use for the supported programming language
- Exit codes to report status in the script
- Formatted output after running the script

You must test the component monitor and script before assigning and using the component monitor within a template as is. Testing the script generates the required database tables for metrics output and verifies output processes.

Script Arguments field

Use this field in the Edit Script window to specify arguments to pass to the script. You may include the variables `{IP}` or `{Node.DNS}`, `{USER}`, and `{PASSWORD}`, which are replaced respectively by the IP address of the target node, the credential user name, and the credential password.

Script Body field

Use the Script Body field to enter your script code. You can write and compile the script in a language you prefer then copy and paste the code into this field. For example, customers have used Python. Depending on the language, you may need to take additional steps for the script and command line to execute.

Scripts must report their status by exiting with the appropriate exit code. The exit code is used to report the status of the monitor, displayed in the Orion Web Console.

Script Engine

This field allows you to specify the scripting language to be used. The default value is `vbscript`. Below is a list of scripting engines that the Windows Script Host supports:

<table>
<thead>
<tr>
<th>Name</th>
<th>File Extensions</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>VBScript</td>
<td>.vbs</td>
<td>Installed by default</td>
</tr>
<tr>
<td>JScript</td>
<td>.js</td>
<td>Installed by default</td>
</tr>
<tr>
<td>PerlScript</td>
<td>.pls</td>
<td>Freeware</td>
</tr>
<tr>
<td>ooRexxScript</td>
<td>.rxs</td>
<td>Freeware</td>
</tr>
</tbody>
</table>
### Custom Template Guide: Server & Application Monitor

<table>
<thead>
<tr>
<th>Name</th>
<th>File Extensions</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>PythonScript</td>
<td>.pys</td>
<td>Freeware</td>
</tr>
<tr>
<td>TclScript</td>
<td>.tcls</td>
<td>Freeware</td>
</tr>
<tr>
<td>ActivePHPScript</td>
<td>.phps</td>
<td>Freeware</td>
</tr>
<tr>
<td>RubyScript</td>
<td>.rbs</td>
<td>Freeware</td>
</tr>
<tr>
<td>Object Rexx engine</td>
<td></td>
<td>Commercial</td>
</tr>
<tr>
<td>Delphi scripting engine</td>
<td></td>
<td>Commercial</td>
</tr>
</tbody>
</table>

VBScript runs locally on the Orion server only. PowerShell uses WINRM that runs over TCP ports 5985 and 5986. All Linux, Unix, and Nagios script monitors use SSH over TCP Port 22.

---

## Create a Windows Script Monitor

The Windows Script monitor allows you to run custom scripts in SAM to monitor Windows-based computers. Before coding and testing a script, review the [Windows script information and components](#) and [best practices](#) for SAM.

You can create a Windows Script monitor to have SolarWinds SAM run a script using Windows Script Host. Windows Script Host comes with VBScript and Jscript, but can be extended with other scripting languages.

Scripts run on the Orion server and use the credentials you specify. The script must both return an exit code and output a text string containing a statistic value conforming to the specifications described later in this section.

Format the Statistic value to use the same decimal separator as the Orion server. The Orion server uses the decimal separator set by its Microsoft Windows regional settings. You may need to log in with an administrator account to perform this action.

1. Click Settings > All Settings > SAM Settings and click Manage Templates.
   - The Manage Application Monitor Templates page opens. All available out-of-the-box and custom templates display.
2. Search to locate, select, and edit an existing template or click [Create New Template](#).
3. Click Add Component Monitor(s) and choose Manually Add Component Monitors.
5. Enter Script Arguments to run for the script. Include the variables \${IP} (or \${Node.DNS}), \${USER}, and \${PASSWORD}, which are replaced with the IP address of the target node, the credential user name, and the credential password.

6. Depending on the script type, the command line may require additional information and parameters such as the file path or hostname.

7. Click Edit Script to enter and test the script.

8. Test the script by selecting settings for the output, specified node, and specified credentials. Click Get Script Output.

Output results display using the script code with success or failure and any additional notes.

9. Click Get Script Output to generate the output and click Save.

The metrics save to outputs to configure as part of the component monitor. These outputs display at polling intervals for the template.

10. When complete, click Submit to save the template.

When tested, output generates into customizable sections. Each output section is named by the Unique ID with a Display Name you can edit.

For each output, you can optionally:

- Convert the value using selected formulas.
- Configure the statistic thresholds to refine alerting for the component monitor. Set the warning and critical thresholds with specific values or baseline data.
- Select a roll-up status sort option to display the best or worst status up through the template and component monitor.
- Add notes to describe the script output.
Adapt an existing Visual Basic script to a Windows Script Monitor in a new template

1. In the Orion Web Console, click Settings > SAM Settings > Create a New Template, then name the template.
2. Click Add Component Monitor, then expand the Custom Component Monitors group, and then check Windows Script Monitor.
3. Click Submit, and then select credentials with the appropriate permissions to run the script on the Orion server, and that also has appropriate permissions to do whatever else the script requires.
4. Copy the Visual Basic script into the Script Body field.
5. Type any script arguments into the Script Arguments field.
6. Specify the critical and warning thresholds, then click Submit.

Macros for Script Arguments

Specify script arguments in the Script Arguments field if needed. You can use the following variables as script arguments:

**${IP}**

This is replaced with the target node's IP Address.

**${USER}**

This is replaced with the user name from the credential set.

**${PASSWORD}**

This is replaced with the password from the credential set.

Script exit codes and text output

Scripts must report their status by exiting with the appropriate exit code, which is then used to report the status of the monitor in the Orion Web Console, as described in Report status through exit codes. For example, to inform SolarWinds SAM that a VBScript reports Up status, you would exit the script using code similar to the following, where 0 reports Up: `Wscript.quit(0)`
Scripts report additional details by sending text to the script's standard output. See [Scripts with text output](#) for details.

Example scripts

Below is a sample vbscript that returns two values:

- The total number of files in a folder
- Twice the total number of files in the same folder

To use this script, copy and paste the following code into the Script Body field. In the Scripts Arguments field, type in C:\Windows, or any other folder you want to monitor.

```vbscript
Option Explicit
On Error Resume Next
Dim lstArgs, path, fso, objDir, objFiles, objFiles2
Set lstArgs = WScript.Arguments
If lstArgs.Count = 1 Then
    path = Trim( lstArgs( 0 ) )
Else
    WScript.Echo "Message: Usage: wscript.exe filelist.vbs [pathToFiles]"
    WScript.Echo "&amp;vbCRLF &amp;"[pathToFiles] Local or UNC Path"
    WScript.Echo "Statistic: 0"
    WScript.Echo "Message: Usage: wscript.exe filelist.vbs [pathToFiles]"
    WScript.Echo "&amp;vbCRLF &amp;"[pathToFiles] Local or UNC Path"
    WScript.Echo "Statistic: 0"
    WScript.Quit( 1 )
End If
Set fso = Wscript.CreateObject( "Scripting.FileSystemObject" )
If fso.FolderExists( path ) Then
    Set objDir = fso.GetFolder( path )
    If( IsEmpty( objDir ) = True ) Then
        WScript.Echo "Message: Object Not Initialized"
        WScript.Echo "Statistic: 0" WScript.Quit( 1 )
    End If
    Set objFiles = objDir.Files
    If( IsEmpty( objFiles ) = true) Then
        WScript.Echo "Message: Object Not Initialized"
        WScript.Echo "Statistic: 0"
        WScript.Quit( 1 )
    End If
    WScript.Echo "Message.Total: " &amp; CInt( objFiles.Count ) &amp; " files in this folder."
    WScript.Echo "Statistic.Total: " &amp; CInt( objFiles.Count )
    WScript.Echo "Message.Twice: " &amp; CInt( objFiles.Count*2 ) &amp; " = twice the number of files in this folder."
    WScript.Echo "Statistic.Twice: " &amp; CInt( objFiles.Count*2 )
```
There are several examples of Windows Script component monitors included in templates. These include: File Count, File Modified, LDAP Connection Monitor, Run 3rd Party Application, and Windows Event Log Count.

Sample scripts are installed on your Orion server, in the folder: C:\Program Files\SolarWinds\Orion\APM\SampleScriptMonitors\WindowsScripts