© 2020 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
  Create a template with the Component Monitor Wizard ........................................................................ 7
  Add component monitors ........................................................................................................................... 8
  Assign the template to nodes ...................................................................................................................... 10

Use script component monitors in SAM ...................................................................................................... 11
  Write scripts for SAM script monitors ................................................................................................... 12
  Test scripts in SAM script monitors ...................................................................................................... 16

Windows scripting in SAM component monitors ....................................................................................... 18
  How the Windows script works .............................................................................................................. 19
  Elements of a Windows script .................................................................................................................. 19
  Create a Windows Script Monitor ......................................................................................................... 20

PowerShell scripting in SAM component monitors .................................................................................... 25
  Elements of a PowerShell script ............................................................................................................ 25
  How Execution Mode impacts PowerShell scripts in SAM .................................................................... 26
  Create a Windows PowerShell script monitor in SAM ............................................................................. 30

Use case: Create a SAM template that uses the Windows PowerShell Monitor to detect possible dictionary attacks .................................................................................................................... 32

Linux/Unix scripting in SAM component monitors ..................................................................................... 35
  Linux/Unix script support in SAM .......................................................................................................... 35
  How the Linux/Unix script works .......................................................................................................... 36
  Elements of a Linux/Unix script .......................................................................................................... 36
  Create a Linux/Unix script monitor in SAM ............................................................................................ 37

Nagios scripting in SAM component monitors ........................................................................................... 40
  Nagios support in SAM ....................................................................................................................... 40
  How the Nagios script works ............................................................................................................. 41
  Elements of a Nagios script ............................................................................................................. 41
Use Nagios Plugins and compiled binary files ...............................................................42
Create a Nagios Script monitor ..................................................................................43
Create custom templates in SAM

As described in the [SAM Administrator Guide](#), SAM provides out-of-the-box (OOTB) templates that allow you to monitor over 200 applications. You can use those templates as is, modify them, or create a custom template from scratch to monitor uncommon components or applications that are not included in the templates provided with SAM.

In addition to creating your own templates, this guide explains how to:

- Use script component monitors
- Create a Windows Script Monitor
- Create a Windows PowerShell script monitor in SAM
- Create a Linux/Unix script monitor in SAM
- Create a Nagios Script monitor

For reference, here are some good examples of custom templates available in [THWACK](#):

- SolarWinds MIB Database
- APC UPS monitoring

Sample script monitors are provided in the following default folder on the Orion server: `C:\Program Files (x86)\SolarWinds\Orion\APM\Sample-Script Monitors`.

**Disclaimer:** Any custom templates, scripts or other content posted in THWACK are provided as a suggestion or recommendation for your internal use. This is not part of the SolarWinds software purchased from SolarWinds, and the information set forth herein may come from third party customers. Your organization should internally review and assess to what extent, if any, such custom templates or scripts will be incorporated into your environment. Any custom content obtained on THWACK is provided to you “AS IS” without indemnification, support, or warranty of any kind, express or implied. You elect to utilize custom templates or scripts at your own risk, and you will be solely responsible for the incorporation of the same, if any.
Here is an overview of the steps involved in creating custom templates:

<table>
<thead>
<tr>
<th>Create a template</th>
<th>To create a template with the Component Monitor Wizard:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1. Navigate to Settings &gt; All Settings &gt; SAM Settings.</td>
</tr>
<tr>
<td></td>
<td>2. Under Getting Started with SAM, click Component Monitor Wizard.</td>
</tr>
<tr>
<td></td>
<td>3. Follow on-screen instructions to add and customize component monitors for the template, and then assign the template to target nodes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>To copy an existing template to create a new template:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate to Settings &gt; All Settings &gt; SAM Settings &gt; Manage Templates.</td>
</tr>
<tr>
<td>2. On the Manage Application Templates page, click the Application Monitor Templates tab.</td>
</tr>
<tr>
<td>3. Select an existing template and click Copy.</td>
</tr>
<tr>
<td>4. After the &quot;Copying application templates&quot; message appears, type &quot;Copy of&quot; in the Search field.</td>
</tr>
<tr>
<td>5. Select the copied template, click Edit, modify as necessary, and save it.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>To create a blank template and add component monitors manually:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate to Settings &gt; All Settings &gt; SAM Settings &gt; Create a template.</td>
</tr>
<tr>
<td>2. On the New Template page, provide a name, description, and other details.</td>
</tr>
<tr>
<td>3. Click Add Component Monitors to add each component monitor you want to include.</td>
</tr>
<tr>
<td>4. Click Submit to save your changes.</td>
</tr>
</tbody>
</table>

| You can also import a template from THWACK and customize it. |

<table>
<thead>
<tr>
<th>Add component monitors</th>
<th>With the base component monitors added to a new template, add other component monitors as needed per application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign a template to a node</td>
<td>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.</td>
</tr>
</tbody>
</table>
Create a template with the Component Monitor Wizard

The fastest method to create a template with a set of component monitors is with the Component Monitor Wizard. The wizard provides the most common and popular component monitors per platform.

1. Click Settings > All Settings > SAM Settings.
2. Under Getting Started with SAM, click Component Monitor Wizard.
3. On the first tab of the wizard, select a component monitor type to add to your template, such as Windows Service Monitor, and then click Next.

4. To select a Target server, enter an IP address or browse to an existing node. Provide all required information and click Next to display a list of available processes, services, and performance counters for the server.
5. On the Select Component tab, select the services you want to add as component monitors for the template, and then click Next.
6. On the Edit Properties tab, customize each component monitor by selecting its check box and clicking [+].
When finished customizing component monitors, click Next.

7. On the Add to Application Monitor or Template tab, select New Application Monitor Template, enter a Name for the template, and click Next.

Do not begin or end a Name with a space. Otherwise, application monitors based on the template may not appear in widgets. Edit the name to remove extra spaces.

8. Assign the template to nodes in your environment according to server type, such as Windows.

Expand and select one or more servers to assign the template.

9. Review the template and click Submit to save it.

After you create a template, you can add more component monitors, as described next.

Add component monitors

When editing a template, you can add component monitors using either of these options:

- **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
This scenario uses the Manually Add Component Monitor option.

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

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

3. Select a component monitor, enter a Quantity based on how many copies you want, and click Add.

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

   - As you work, click Save and Continue Working to save changes.

5. When finished adding component monitors, click Submit.
Assign the template to nodes

To begin monitoring with the template, assign it to one or more nodes, thus establishing an application monitor on each node.

1. 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.
Use script component monitors in SAM

Script component monitors (also called "script monitors") offer limitless options for monitoring and returning metrics for target servers. Each monitor has different options to execute the script, enter credentials, set working directories, and then generate and display returned values as output.

SAM includes the following predefined script component monitors, but you can also write your own.

- Windows PowerShell Monitor
- Windows Script Monitor
- Linux/Unix Script Monitor
- Nagios Script Monitor

For details on creating script monitors, see:

- Write scripts for SAM script monitors
- Test scripts in SAM script monitors
- Windows scripting in SAM component monitors
- PowerShell scripting in SAM component monitors
- Linux/Unix scripting in SAM component monitors
- Nagios scripting in SAM component monitors

Note the following details about SAM script monitors:

- Before writing scripts, review Best practices for SAM templates, application monitors, and script monitors.
- Sample script monitors are available on the Orion server at: C:\Program Files (x86)\SolarWinds\Orion\APM\SampleScriptMonitors
- When using SolarWinds macros, consider assigning them to named variables in your scripts.
- To learn about using data transformations, see these online help topics:
  - Conversion value
  - Available data transformations for SAM monitors
- The SAM forum in THWACK also provides content about scripting.
**Disclaimer:** Scripts provided outside of the Orion Web Console are not supported under any SolarWinds support program or service. Scripts are provided AS IS without warranty of any kind. SolarWinds further disclaims all warranties including, without limitation, any implied warranties of merchantability or of fitness for a particular purpose. The risk arising out of the use or performance of the scripts and documentation stays with you. In no event shall SolarWinds or anyone else involved in the creation, production, or delivery of the scripts be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use of or inability to use the scripts or documentation.

---

### Write scripts for SAM script monitors

Every script monitor accepts code through the Script Body field that you can access by expanding Component Monitor details, as shown in this example of a Linux/Unix Script Monitor included in the OpenLDAP template:

![Linux/Unix Script Monitor](image)

You can develop code against the target system directly in the SAM script monitor or using your preferred scripting environment and programming language, and then copy it into the Script Body field. For example, Microsoft provides a Windows PowerShell Integrated Scripting Environment (ISE). Programming environments and applications give you script libraries and color-coded source code that may be easier to use than the Script Body field.
Use code comments to keep detailed steps and responses in your code. If additional administrators work in the script monitors, the comments provide context for the code. For additional tips, see [Script best practices](#).

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, test the script to verify credentials and target server access, as well as generate output.

To optimize 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 exit codes and output

When a script runs, SAM looks for two main things:

- **An exit code** that indicates if the script ran successfully. That value is used to report the status of the monitor in the Orion Web Console. For example, to inform SAM that a VBScript reports Up status, exit the script using code similar to the following, where 0 reports Up: `Wscript.quit(0)`. See [Report status through exit codes in SAM script monitors](#).

- **Script output** that consists of returned metrics, as described next.

Scripts can also report additional details by sending text to the script’s standard output. See [Scripts with text output](#) for details.

For script output, SAM is mainly interested in the ‘Statistic’ value followed by a numerical value (for example, `Statistic: 123:`). How that information is reported back to SAM and appears in the Orion Web Console depends on the scripting language that you’re using.

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

SAM expects 1 to a maximum of 10 metrics returned with a string of Statistic and a numerical value and an optional message. Each returned metric generates an output. When returning multiple metrics, you can add descriptive labels to the returned statistics, such as `StatisticFileSize`, `StatisticMailboxCount`, and `StatisticName`. 
If you exceed the maximum number of allowed output pairs of ten, the remainder above the tenth output pair is ignored.

Note the following details about statistics:

- If using labels for statistics, do not use spaces.
- If you delete the output, the metric won’t appear in the Orion Web Console, but SAM still saves it to the Orion database.
- For Windows scripts, format the Statistic value to use the same decimal separator as the Orion server, as determined by Windows regional settings.

Report status through exit codes in SAM script monitors

As described in Use script component monitors in SAM, scripts should report their status to SAM by exiting with the appropriate exit code which results in an Up (0), Warning (2), or Critical (3) status. The exit code determines what appears for the component in the Orion Web Console.

When an exit code is received, SAM creates an appropriate dynamic evidence table structure 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.

<table>
<thead>
<tr>
<th>Exit Code</th>
<th>Service State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Up</td>
</tr>
<tr>
<td>1</td>
<td>Down</td>
</tr>
<tr>
<td>2</td>
<td>Warning</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
</tr>
<tr>
<td>Any other value</td>
<td>Unknown, for example 4</td>
</tr>
</tbody>
</table>

The following code snippet highlights proper usage of exit codes:

```powershell
if ($Error.Count -eq 0) {
    Write-Host "Message: User $address received: $stat items during last month";
    Write-Host "Statistic: $statitem";
    Exit 0;
}
Write-Host "Message: $($Error[0])"
Exit 1;
```

The two exit codes in this example are conditional:

- When Exit 0; (status of Up) is reported, the message and statistic are displayed and the monitor shows a status of Up.
When Exit 1; (status of Down) is reported, the message and statistic are not displayed and a status of Down is reported.

For example, if you want a script to inform SAM that a PowerShell script reports an Up status, exit the script using Exit 0.

**Scripts with text output**

As described in Use script component monitors in SAM, SAM looks for two main things when a script return values for target servers:

- **An exit code** that indicates if the script ran successfully. That value is used to report the status of the monitor in the Orion Web Console. See Report status through exit codes in SAM script monitors.
- **Script output** consists of returned metrics that generate an output, which is then displayed in the Orion Web Console.

Scripts can also report additional details by sending text to the script’s standard output, such as informational text or error details. Depending on your needs, you can create different If/Then/Else statements for the statistic value that each have a different Message format so you know what your script is doing.

💡 **Want to see how other SolarWinds customers work with scripts?** Visit the Server & Application Monitor forum in THWACK.

Note the following details about text output:

- There is a limit of ten Statistic and Message pairs per script. Each Statistic is a numeric value; each Message is readable text.
- Statistic and Message pairs can be placed anywhere in the script output.
- The Statistic and Message names you give must contain valid letters and/or numbers.
- Named pairs can be dynamically defined in the code, but SAM expects the output to remain consistent.

Additionally, the Message variable use the same value as your Statistic variable (for example, Statistic.value1 and Message.value1, Statistic.value2 and Message.value2). If your Message.<name> does not match your Statistic.<name>, then SAM will throw an error.
<table>
<thead>
<tr>
<th>Detail Type</th>
<th>Required</th>
<th>Meaning</th>
</tr>
</thead>
</table>
| Statistic   | Yes      | 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 |
| Message     | No       | 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 |

Sample output:

```
# Script comment: This shows two pairs. Ten pairs are possible.
Statistic.CPU: 31.08Message.CPU: svchost.exe cpu usage
Statistic.RAM: 1234.56Message.RAM: svchost.exe ram usage
```

Test scripts in SAM script monitors

When configuring script component monitors in SAM, test scripts before you verify credentials and access. When the test completes, SAM registers each returned metric as a numbered output in the Orion database. Until they're tested, scripts and component monitors return an initial Unknown status. After testing, polling returns accurate application status.

Verify you have the correct credentials with assigned account permissions to execute scripts on the Orion server and target servers. The script monitors may provide fields for credentials, or you may need to provide credentials in the script code, arguments, or command line.
To test a script monitor:

1. Navigate to the page for the template or application monitor.

2. Scroll down to the list of component monitors, as shown here:

<table>
<thead>
<tr>
<th>Component Monitor Name</th>
<th>Component Type</th>
<th>Test Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operations Rate per Sec</td>
<td>Linux/Unix Script Monitor</td>
<td>Will test against ENG-AUS-SYS-477</td>
</tr>
<tr>
<td>Active Operations</td>
<td>Linux/Unix Script Monitor</td>
<td>Test node not specified</td>
</tr>
</tbody>
</table>

3. Select the script monitor.

4. Click Set Test Node to select a node.

5. Click Test.
Windows scripting in SAM component monitors

The 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. It does include details about coding in various programming languages.

Before coding and testing a script, review Create custom Windows scripts and Best practices for SAM templates. Sample script monitors are available on the Orion server at: C:\Program Files (x86)\SolarWinds\Orion\APM\SampleScriptMonitors

To display a list of out-of-the-box SAM templates that support Windows scripting:

- Navigate to the Manage Templates page.
- Click Settings > All Settings > SAM Settings > Manage Templates.
- On the Manage Application Monitor Templates page, enter Windows in the search field.

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. Verify that the install location for the scripting language is in the system environment path variable.
3. In the SAM template, expand the details for the Windows Script Monitor and specify the scripting language in the Script Engine field.
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.

Elements 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 verify 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

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. 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. For details, see Report status through exit codes in SAM script monitors.

**Script Engine**

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>
<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>

Note the following details:

- If the application monitor is set to Agentless, the VBScript runs on the Main Polling Engine (usually the Orion server).
- If the application monitor is set to Agent, the script runs on the target node.
- PowerShell uses WinRM that runs over TCP ports 5985 and 5986.
- All Linux, Unix, and Nagios script monitors use SSH over TCP Port 22.

💡 **Want to see how other SolarWinds customers use script monitors?** Visit the Server & Application Monitor forum in THWACK.

**Create a Windows Script Monitor**

You can use the predefined Windows Script Monitor included in SAM to run custom scripts that use Windows Script Host to monitor Windows-based computers. Windows Script Host comes with Visual Basic script (VBScript) and JScript, but can be extended with other scripting languages.
Before coding and testing a script, review [Create custom Windows scripts](#) and [Best practices for SAM templates](#).

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.

1. Click Settings > All Settings > SAM Settings and click Manage Templates.
2. When the Manage Application Monitor Templates page appears, 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.

   - 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 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.

![Script Output #1](image)

Adapt an existing VBScript to a Windows Script Monitor in a new template

1. Click Settings > SAM Settings > Create a New Template, and 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 VBScript 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 SAM that a VBScript reports Up status, 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**

Following 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]
&amp;vbCRLF &amp;"[pathToFiles] Local or UNC Path"
WScript.Echo "Statistic: 0"
WScript.Echo "Message: Usage: wscript.exe filelist.vbs [pathToFiles]
&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 )
WScript.Quit( 0 )
Else
WScript.Echo( "Message: Folder Not Found" )
WScript.Echo "Statistic: 0"
WScript.Quit( 1 )
End If

Several sample scripts are installed in the following default folder on the Orion server:
C:\Program Files (x86)\SolarWinds\Orion\APM\SampleScriptMonitors\WindowsScripts
PowerShell scripting in SAM component monitors

As described in Use PowerShell in SAM, many features, such as AppInsight for IIS and AppInsight for Exchange, leverage PowerShell to execute commands and gather information from remote, target systems. Several out-of-the-box SAM templates also use the predefined Windows PowerShell Monitor to gather data.

The ability to create and deploy PowerShell scripts to remote machines within SAM is a powerful advantage for system administrators. With an interactive prompt and scripting environment, PowerShell provides access to the file system on remote computers, along with data stores such as the registry so you can use it monitor applications that might not otherwise be monitorable.

In addition to this overview, see the following topics in this section:

- Create a Windows PowerShell script monitor in SAM
- How Execution Mode impacts PowerShell scripts in SAM
- Use case: Create a SAM template that uses the Windows PowerShell Monitor to detect possible dictionary attacks

**Disclaimer:** SolarWinds recommends that you always review scripts to check for malicious code. Custom scripts you create or download from 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.

Elements of a PowerShell script

Each PowerShell script should include the following elements:

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

Avoid using Clear-Host methods in PowerShell scripts. Click here for details.
Here is a summary of what happens when a Windows PowerShell script runs in SAM:

1. The Windows PowerShell Monitor checks if the Execution Mode is set to Local Host or Remote Host.
   - **If Local Host**, the script executes using the script arguments on the Orion server.
   - **If Remote Host**, the script connects via SSH connection to run the script on the target server.

   WinRM must be properly configured on the Orion server and target servers so scripts can run. See Use PowerShell in SAM.

2. The script executes and collects metrics from the target server using entered credentials.

3. The script reports its status to SAM by finishing with an exit code to indicate if the monitor is in an Up (0), Warning (2), or Critical (3) status.

4. SAM parses the text output, saves data, and reports values using output formats from the component monitor. The monitor can return up to ten pairs — 10 statistic values and 10 optional messages. If you exceed the maximum allowed, remove the excess output pairs or they will be ignored.

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

**How Execution Mode impacts PowerShell scripts in SAM**

Whether using the predefined Windows PowerShell Monitor on its own or within a template, you can configure the Execution Mode to control how and where PowerShell scripts run – either locally on the Orion server, or remotely on target servers, with or without required credentials.
Leave Execution Mode set to the default value, Local Host, to run scripts locally from the Orion server. The script will run as a service under the local system account unless you also enable the "Run the script under specified account" option. If you need to run specific non-default cmdlets for the target server, install the needed cmdlets on the Orion server.

Set Execution Mode to Remote Host to execute scripts on the selected target node. SAM will create a remote session via WinRM on the target server, copy the script to the target server, and then run the script there.

WinRM must be properly configured on the Orion server and target server so scripts can run. See Use PowerShell in SAM.

The following options are available for Remote Host mode:

- **Use HTTPS Protocol**: The default value is HTTP. Select HTTPS if you want the monitor to send and receive encrypted WS-Management protocol requests and responses for increased security.
- **URL Prefix**: Specify the URL prefix on which to accept HTTP or HTTPS requests. The default is wsman.
- **Port Number**: Specify the TCP port used to listen for traffic.
  - For WinRM 1.1 and earlier, the default port is 80.
  - For WinRM 2.0, the default port is 5985.

**Run scripts under a specified account**

SAM allows you to run PowerShell scripts using the local impersonation of a specified user account (for example, an account with domain privileges). By enabling the "Run the script under specified account" option when using Local Host as the Execution Mode, you can control what the script can and cannot do via the permissions assigned to the account you specific in the Credential for Monitoring field.

The following table describes how SAM handles credentials for a PowerShell script based on:

- The selected Execution Mode: Local Host or Remote Host
- If the "Run the script under specified account" option is enabled.
<table>
<thead>
<tr>
<th>Execution mode</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Host</td>
<td>Disabled</td>
</tr>
<tr>
<td>Local Host</td>
<td>Enabled</td>
</tr>
<tr>
<td>Remote Host</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Make sure that:

- WinRM is enabled and properly configured on the Orion server and target servers.
- The credential has rights to log into the Orion server.
- The credential has sufficient rights on the target server to perform tasks in the script.

To increase security, SolarWinds recommends configuring WinRM to use HTTPS. See [PowerShell Remoting Security Considerations](https://docs.microsoft.com) (© 2018 Microsoft Corp., available at [https://docs.microsoft.com](https://docs.microsoft.com); link obtained on October 19, 2018).

To troubleshoot Local Host Execution Mode:

- Make sure the file exists in the path specified in the script.
- Verify that the user account specified in the credentials section of the component monitor is a local administrator with sufficient permissions to execute the PowerShell script. Try logging into the server using those credentials and running the script manually.
SAM executes PowerShell scripts using 32-bit PowerShell by default. If a PowerShell script runs fine in the Orion Web Console but causes errors in logs, try changing the "Platform for polling jobs" from x86 to x64. See PowerShell Script Monitor Running in Polling Job on x86/x64 Platform in THWACK.

Example

The following script can be used in the Orion Web Console, or in the PowerShell console. It returns 0 as the exit code and the Hostname of the Orion server (Local Host) or the Hostname of the target machine (Remote Host).

In the Windows PowerShell console, the script returns the local machine Hostname. If, for some reason, the script cannot get the hostname, it returns 1 as the exit code and a message of “Host not found.”

```powershell
$stat = $env:computername;
if ($stat -ne $null)
{
    Write-Host "Statistic: 0";
    Write-Host "Message: $stat";
}
else
{
    Write-Host "Statistic: 1";
    Write-Host "Message: Host not found";
}
```

Tips

To save a script as a PowerShell (ps1) file in Notepad, save it as Get-Date.ps1, and then rename the file from *.txt to *.ps1.)

To open the PowerShell console:

- For Windows x64:
  - For x64, navigate to Start > Accessories > Windows PowerShell > Windows PowerShell
  - For x86, navigate to: Start > Accessories > Windows PowerShell > Windows PowerShell (x86)
- For Windows x86:
  - Navigate to Start > Accessories > Windows PowerShell > Windows PowerShell

To run this script via remote execution, use this command:

```powershell
invoke-command -ComputerName SOME_PC -Credential SOME_PC\SOME_USER -ScriptBlock { Get-Date }
```
To run this script via local execution:

1. Name the file `Get-Date.ps1`.

2. Double-click the `ps1` file.

**Create a Windows PowerShell script monitor in SAM**

As described in [PowerShell scripting in SAM component monitors](#), you can use the [Windows PowerShell Monitor](#) included in SAM to create custom scripts for templates, and then assign templates to the main Orion server or remote target nodes. After running a script, SAM processes the script’s exit code and text output to display results in the Orion Web Console.

**To create a script with the Windows PowerShell Monitor:**

1. Click Settings > All Settings > SAM Settings > Manage Templates.

   The Manage Application Monitor Templates page displays all available out-of-the-box and custom templates.

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 then click add.
5. Enter all arguments and parameters for the Script Arguments. If using Remote Host as the Execution Mode, the Port Number should correlate with the TCP port used by the listener on the remote, target server. The default port for WinRM is 5986.

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

Sample scripts are provided in the following default folder on the Orion server:
C:\Program Files (x86)\SolarWinds\Orion\APM\SampleScriptMonitors

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.

10. Create an application monitor based on the new template and assign it to a target node. See
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 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.

![Script Output](image)

If a PowerShell script monitor causes an exception that references "SetBufferContents", see this [Success Center article](#).

**Use case: Create a SAM template that uses the Windows PowerShell Monitor to detect possible dictionary attacks**

The following use case shows how to create a SAM template that includes a [Windows PowerShell Monitor](#) with a script that tracks the average number of Read operations performed by `lsass.exe` on target systems. `lsass.exe` is a Windows process that validates user logins, which is useful for detecting dictionary attacks.

Here is a high-level overview of what’s involved:

- Create the new template.
- Add the Windows PowerShell Monitor as the primary component monitor in the template. It will execute scripts on target systems to gather data.
- Provide credentials so the script can run on target systems.
• Add a PowerShell script that uses `Get-WmiObject` calls to measure the average `ReadOperationCount` for the `lsass.exe` process.

• Assign the new template to nodes.

Before you begin, make sure the WinRM service is properly configured on polling engines and target servers to support remote execution of PowerShell commands. See Using PowerShell in SAM.

To create a template with a Windows PowerShell Monitor:

1. Verify that the WinRM service is properly configured on the main Orion server and target servers so you can execute PowerShell commands remotely. See Using PowerShell in SAM.

2. Click Settings > All Settings > SAM Settings > Create a New Template

3. Name the template (for example, `Lsass.exe PowerShell Monitor`).

4. Click Add Component Monitor, type "powershell" in the search field, select the Windows PowerShell Monitor, and click Add.

5. Select a Credential for Monitoring with appropriate permissions to run the script on the Orion server, and that also has appropriate permissions to do whatever else the script requires (in this case, to get the average number of read operations performed on the target node).

6. Select Remote Host as the Execution Mode.

7. Click Edit Script and then paste the following PowerShell script into the Script Body field:

   ```powershell
   $avg = Get-WmiObject win32_process -ComputerName '${IP}' -Credential '${CREDENTIAL}' | Where-Object {$_ .Name -eq "lsass.exe"} | Measure-Object -property ReadOperationCount -Average; Write-Host 'Statistic: ' $avg.Average; exit(0)
   ```

   The PowerShell code does the following:

   a. Reads the average `ReadOperationCount` for the process `lsass.exe` running on the server at the IP address specified by the `${IP}` variable, using the credential specified by the `${CREDENTIAL}` variable.

   The user name from the specified Credential for Monitoring is stored automatically in the `${CREDENTIAL}` variable by the monitor. Do not add the `${CREDENTIAL}` variable in the Script Arguments field. When the script runs and needs a password, the monitor automatically gets the password from the Credential for Monitoring.
b. Writes the statistic information gathered by the script.

c. Exits the script with an exit code (0) to report the status of the monitor, as displayed in the Orion Web Console. See Report status through exit codes in SAM script monitors.

8. Enter the following Script Arguments:

Use the token `${IP}` to populate the IP address in the script body with the IP address of the target node. You can then access the value in the script body using the variable `${IP}`.

9. Select Count Statistic as Difference to change the statistic to be the difference in query values between polling cycles.

10. Click Set test node. Browse the tree view, select the desired target node for the PowerShell script, and then click Select.

11. Change the Statistic Warning Threshold to, greater than 800.

12. Change the Statistic Critical Threshold to, greater than 1000.

13. Click Test, and then click Submit.

To assign your new template to a node:

1. Navigate to the Manage Application Template page, select the new template, and click Assign to Node.

2. Expand the tree view, select a target node, and then click Next.

3. Select "Inherit credentials from template", and then click Test to confirm the credentials.

4. Click Assign Application Monitors and then click Done.

5. (Optional) Configure alerts to be notified when spikes occur.
Linux/Unix scripting in SAM component monitors

You can use the Linux/Unix Script Monitor included in SAM to run custom scripts on Linux/Unix systems. You can create multiple instances of this component monitor per template to run different custom scripts.

For a list of templates that use customized instances of the Linux/Unix Script Monitor, 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.

Related topics include:

- Create a Linux/Unix script monitor
- Configure Linux/Unix systems for the Orion agent for Linux

The SAM forum in the THWACK provides content about Linux/Unix scripting. Sample script monitors are available on the Orion server at: `C:\Program Files (x86)\SolarWinds\Orion\APM\SampleScriptMonitors`

Linux/Unix script support in SAM

Prerequisites

Linux/Unix is installed and configured on target servers.

Credentials

Include credentials that can run scripts on the monitored Linux-based computer.

Python installation

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:

```bash
python -c "import sys; from xml.dom.minidom import parseString; print sys.version"
```
See the Python site (© Python Software Foundation, available at https://www.python.org/, obtained on February 5, 2020) 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 it with the Apache template, you need to configure Apache access. For full configuration details for all services, see Configure Linux/Unix systems for monitoring by the Orion agent.

**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.

**Elements 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 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
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 on the target system where the script processes when executed. The default 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, some customers use Python. Depending on the language, you may need add 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, as displayed in the Orion Web Console. For details, see [Report status through exit codes in SAM script monitors](#).

**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 script monitor in SAM

This topic explains how to use the [Linux/Unix Script Monitor](#) included in SAM to run custom scripts in SAM to monitor Linux-based computers. Before coding and testing your script, review [Configure Linux/Unix systems for the Orion agent for Linux](#) and [Best practices for SAM templates](#).

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 Credentials for Monitoring.

6. Enter a Working Directory to process the script, such as `/tmp`.

7. Enter a Command Line to run the script including all parameters using the format `<scripting language path> ${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.

![Image of the script output](image)

Related topics include:

• [Linux/Unix scripting in SAM component monitors](#)
• [Configure Linux/Unix systems for the Orion agent for Linux](#)
• [Linux/Unix Script Monitor](#)
Nagios scripting in SAM component monitors

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 section describes how to use Nagios scripts with SAM, including script arguments and output formats. To learn about the Nagios programming language, see the Nagios website (© 2019, Nagios Enterprises, LLC, available at https://www.nagios.org, obtained on June 25, 2019).

Want to see how other SolarWinds customers use script monitors? Visit the Server & Application Monitor forum in THWACK.

Nagios support in SAM

SAM supports using Nagios scripting with Nagios installed and configured on target servers, as well as Nagios Plugins and compiled binary files (ELF) for Linux-based computers.

For a list of templates that support Nagios scripts and use customized instances of the Nagios script monitor, navigate to the Manage Templates page.

1. Click Settings > All Settings > SAM Settings.
2. Under Application Monitor Templates, click Manage Templates.
3. In the search field, enter Nagios to filter the list.

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

To use this component monitor with Nagios, as polled by Orion agent for Linux, enter a script as follows. For the Script Argument, use `python $(SCRIPT)`.

```python
from random import randint
STAT_PAIRS_COUNT = 10
```
for stat_id in range(STATPAIRS_COUNT):
    print 'Statistic.random%d: %d
Message.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.

Elements of a Nagios script

SolarWinds recommends writing scripts in the Nagios Script Monitor to verify access between the Main Polling Engine (usually the Orion server) and 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 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. See Report status through exit codes in SAM script monitors.

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 a 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 from the Nagios site and compiled binary files (ELF) used on Linux-based computers.

For Nagios Plugins:

- Install plugins 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 these topics:

- Best practices for SAM templates
- Nagios scripting in SAM custom templates


For Nagios plugins:

- Install compiled plugins on the target system. SAM does not transfer plugins.
- 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.

   ![Custom template interface](image)

5. Select the Authentication Type and Credentials for Monitoring.

6. Enter a Working Directory for processing the script, such as `/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.
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 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.