Okay, let me draw up a good use case for ScriptRunner.
You and your colleagues form the IT infrastructure operations
team at your business. Over the past year, your team embraced
PowerShell for administrative automation.

However, you’ve identified several problems with your
team working with their PowerShell scripts individually:

- No centralized management
- Uncontrolled changes to scripts shared among colleagues
- No script test/approval process
- Difficult script discovery (email being the chief, and
  inefficient, culprit)
- Lack of script versioning
- Tough to prevent erratic script usage and accompanying
  execution errors
- Limited transparency (who runs which scripts when)

Today I’d like to introduce you to ScriptRunner. I will explain how
to obtain the software, install it, and configure it to solve all the
above problems.

Oh, before we begin, a quick but important side note:
AppSphere ScriptRunner is not related in any way, shape, or
form to Adaptavist ScriptRunner. I wanted to make this clear be-
cause of Adaptavist’s high search engine ranking. Here’s a protip:
include ‘powershell’ in your web searches for ScriptRunner.
Onward!

PRODUCT INSTALLATION NOTES
You can download an evaluation version of ScriptRunner from
their website, but they don’t make the process as clear as they
could. Instead, AppSphere really wants you to participate in a
live, interactive demo.

Anyway, I followed the ScriptRunner installation instructions and
set up my environment in under one hour. Specifically, I installed
the ScriptRunner service on a Windows Server 2016 domain
member server with the Web Server role already installed.
Configuration choices include selecting a TCP port for the web
service to listen on, designating an Active Directory group for
ScriptRunner admins, and enabling PowerShell remoting on the
server – not a big deal in most cases.
Next, I installed the ScriptRunner Web Apps: you interact with the back-end service with these web front-ends. The note here is that you may want or need to configure the web app to use your company’s internal or public TLS/SSL digital certificate to support HTTPS.

**ADMIN APP SETUP**

Point your web browser to server-name/scriptrunner/admin to load up the ScriptRunner Admin App. I show you the interface next, and I’ll explain each navigational node afterward:

- **Actions:** An Action defines the entire context of a managed PowerShell script
- **Queries:** Create and edit reports that may or may not use a library script for logic
- **Targets:** Your Actions run on these systems – choices include local execution, PowerShell remoting execution, Docker container, Office 365 service, or Azure Resource Manager service
- **Scripts | Cmdlets:** View your centralized PowerShell script library – you need to use PowerShell to change the library location (such as to a Git repository)
- **Credentials:** An internal credential safe for use in your scripts
- **Delegation:** Give non-administrators the ability to run Actions
- **Automation:** Lets you connect ScriptRunner to an external IT service management/monitoring platform, such as ServiceNow or Nagios.
- **Settings:** Verify your license status, tag settings, and ScriptRunner library location.

Let me say that the ScriptRunner Documentation site appeared to be incomplete. As of this writing in February 2019, I could view only the Installing & Updating document. The Configuring and Using Apps guides would have been especially helpful, were they actually present. Yes, I get that AppSphere wants to walk you through the software personally, but I’m a “read the manual” kind of guy, you know?

As I mentioned, you can configure some tasks only via the ScriptRunnerSettings module. Run the following command to see the available commands:

```powershell
Set-APISettings -ScriptLibraryPath 'C:\script-library' -Restart
```

Another big disappointment was when I ran Get-Help against the ScriptRunner cmdlets, and they came up empty. The AppSphere documentation team really has their work cut out for them!

The idea with the central script library is you can point it to a location you’re tracking with Git version control (or another source code version control system). After some experimentation, I found this worked to customize the library store:

In case you need it, the default library path is `C:\ProgramData\AppSphere\ScriptMgr\`. You’re welcome!

Now you can work in the **Scripts | Cmdlets page** of the Admin App. I suggest your team adds their scripts to the library through the Admin App interface so you can add taxonomic tags as shown in the following composite screenshot.

You will also want to mosey over to the **Credentials page** and populate the table with any domain and/or local credentials for use with your scripts. By default, ScriptRunner uses Windows Credential Manager on your ScriptRunner service host server. Alternatively, you can use an external password server such as Thycotic Secret Server or CyberArk Password Vault. Finally, it’s a good idea to populate the **Targets** table with the remote servers and services that will be the target of your PowerShell administration scripts. Again, you can assign taxonomic tags to these resources to make your queries and reports more effective.
CONFIGURING ACTIONS
Okay, let’s go to the Action where you tie all the aforementioned ScriptRunner resources together. Let’s say, for example, you have a PowerShell script that gathers disk metrics from one or more hosts defined by a parameter value.

You can create an action that runs that script on a schedule, supplying particular parameter values to send a notification, and optionally, delegated personnel (for instance, support staff) can run it ad hoc.

In the Admin App, go to Actions and click Create. Building a new action requires supplying the following information:

- assign parameter values at runtime
- add taxonomic tags and descriptions
- select target(s)
- customize PowerShell execution properties (module loads, timeout values, etc.)
- configure email notification if you specified an SMTP server in your settings
- define a schedule
- delegate the action to a sub-administrative group or identity

You can then use the bottom toolbar in Admin App to customize pre-existing actions as shown in the next screenshot.

DELEGATE APP SETUP
One cool selling point of ScriptRunner is you can control who can run your centrally managed PowerShell scripts. A delegation is authorized either by Active Directory, a local identity store (on a workgroup computer, for instance), or through claims-based single sign-on (SSO) identity. Specifically, your delegation gives a sub-administrative identity the permission to run one of your stored scripts or actions without having to elevate their permissions. This feature supports DevOps and today’s self-service cloud computing lifecycle.

The idea is that you as administrator create your Actions and Delegations, and your support desk executes those script tasks through the ScriptRunner Delegate App, found at the path servername/scriptrunner/delegate.

As you would expect I hope, the delegated user sees only those scripts and actions to which he or she was granted access. The delegated user can either choose from pre-created (ValidateSet) parameter values or specify them at runtime. You can see an example of the interactive form in the next screenshot.
REPORTING CAPABILITY
Clicking Report from the bottom toolbar brings up a pane that lists operational details in reverse chronological order. Although the reports do give the basic “who did what, when” details and allow you to download its contents in .txt format, I found that:

- There was no search capability
- It listed the report data in document report format rather than in an easily sortable/scannable tabular format

That said, I suggest that AppSphere make the following changes to make ScriptRunner easier to user and more effective:

- Complete the product documentation, especially for the ScriptRunner PowerShell module
- Make it easier to specify the library location, particularly for use with Git or another version control system
- Present reports in tabular format and make them easily filterable and searchable
- Have the Admin App and Delegate App open to a tutorial dashboard with easy 1-2-3 config steps to help ease everyone’s learning curve

I would give you licensing and pricing details, but that information isn’t available on the ScriptRunner website either. You can contact AppSphere at your convenience though. General licensing information can be found at [https://www.scriptrunner.com/en/licensing/](https://www.scriptrunner.com/en/licensing/)

I hope reading this review has you thinking more about important IT operations questions like “How can our team reduce repetitive efforts and increase both security and efficiency with our PowerShell scripting?” At base, I think AppSphere ScriptRunner is a pretty good product for this purpose.

Timothy Warner is a Microsoft Cloud and Datacenter Management MVP
This article was also published on 4sysops.com in February 2019.