Cross Platform PowerShell remoting

BY

CHRISTOPH BERGMEISTER

> whoami

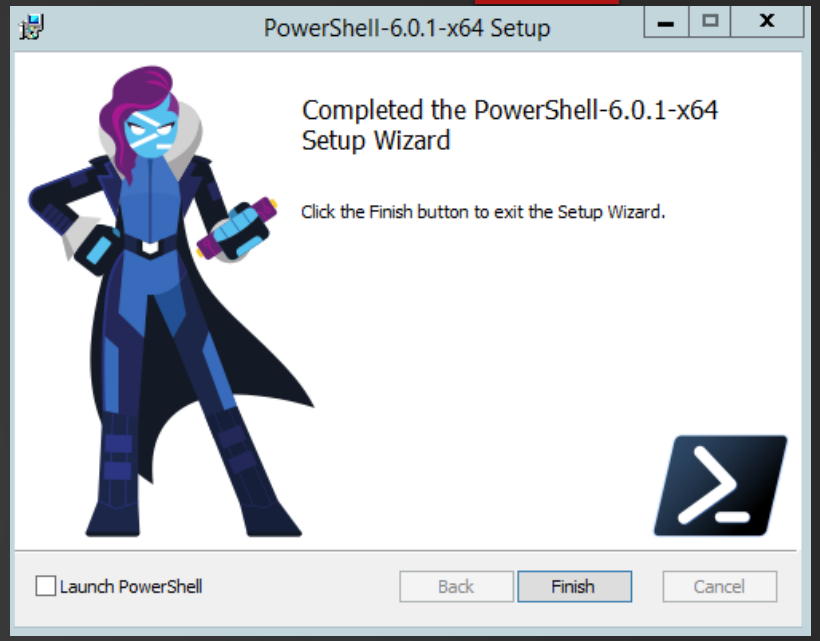
- Christoph Bergmeister
- MSc. Computational Physics, then in Software for 3 years now
- Full stack (Asp) .Net (Core) with a DevOps focus (PowerShell/Azure/VSTS)
- Author of posh-dotnet and posh-vsts-cli
- Maintainer of PSScriptAnalyzer and contributor to the PowerShell Core repo

- GitHub: https://github.com/bergmeister/
- Twitter: https://twitter.com/ChristophBergm
- Blog: https://bracketsandbraces.blogspot.co.uk/

What is PowerShell 6.0 (aka PowerShell Core)?

- Cross Platform: common Linux distros, macOS, Docker and also on ARM systems (raspberry-pi)
- Self contained (zip/tar-ball) or side-by-side installation
- Improved engine and cmdlets
- More breaking changes than usual (but mostly for the better) were allowed due to porting to .Net Core
- Limited functionality/cmdlet coverage compared to Windows PowerShell 5.x
- New name of main binary: powershell -> **pwsh**
- Modern release cycle (FFO every 6 months)
- Daily build: iex "& { $(irm https://aka.ms/install-pscore) } -Daily"

My (34) contributions to PowerShell



6.0

- ConvertFrom-Json **-AsHashTable** -> see my recent blog
- MSI installer improvements
- Internal Build and documentation related stuff

6.1

- More installer features: explorer.exe context menu, Enable-PSRemoting, etc.
- (still) active:
  - Location history: 'cd –'
  - New-TemporaryFile **–Extension** -> pending on corefx

My (23) contributions to PSScriptAnalyzer

- Invoke-ScriptAnalyzer **-Fix**
- Invoke-ScriptAnalyzer **–EnableExit**
- Upgrade build to netcore2.0
- Building and debugging with Visual Studio
- Warn against usage of (read-only) automatic variables
- Various bug and documentation fixes
- In progress: Warn against incorrect usage of ‘=‘, ‘==‘, ‘>’ in if/while statements

Cross Platform remoting

Scenario

- .Net Core development with CI system tests against Windows/Linux
- -> How to handle remote scripting for deployment, setup, etc?

Solution

- PowerShell Core + Win32OpenSSH
- -> Invoke-CommandCrossPlatform -HostName $FQDN -UserName $userName -ScriptBlock $scriptBlock

Disclaimer

- Windows port of OpenSSH is still in **Beta** despite the version number recently being bumped to 1.0 due to it being shipped in Windows.
- Do not use the new 1.x version shipped as part of Windows, it's much more limited! -> Use 0.0.24 or RTFM

Blog: https://blogs.technet.microsoft.com/heyscriptingguy/2017/12/29/cross-platform-powershell-remoting-in-action/

How it works in CI end-to-end