

Why I love PowerShell Desired State Configuration and you should as well

Nicholas Dille, RDS MVP

E2EVC Berlin, 12.-14.06.2015



Who is this guy?

Husband, father, geek, author, Aikidoka

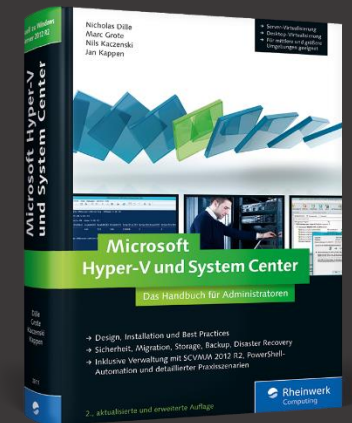
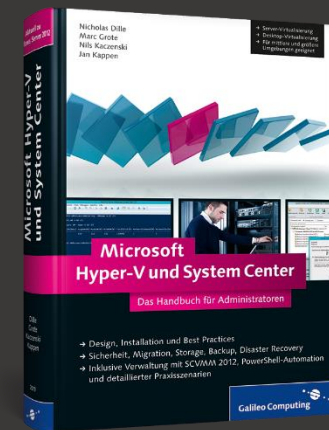
Teamlead Enterprise Consulting @ Makro Factory

10+ years of experience in SBC/VDI market

RDS MVP since 2010

<http://dille.name/blog>

@NicholasDille



Microsoft Hype-V und System Center
in Windows Server 2012 (R2)

Agenda

Design Goals



Style



Security



Custom Resources



Now what?



Desired State Configuration

PowerShell is imperative

Example

```
$f = Get-WindowsFeature -Name Telnet-Client  
If (-not $f.Installed) {  
    Add-WindowsFeature -Name Telnet-Client  
}
```

Code is your responsibility

- Reboot handling
- Error handling

PSDSC is declarative

Example

```
WindowsFeature Telnet-Client {  
    Name = 'Telnet-Client'  
    Ensure = 'Present'  
}
```

Code is provided in resources

- Works across reboots
- Support for dependencies

Properties

Idempotent

„[...] the property of certain operations in [...] computer science that can be applied multiple times without changing the result beyond the initial application.“ – [Wikipedia](#)

Idem + potence = same + power

Monotonic

Order is preserved based on dependencies

Minimalistic

Resource should be limited to serve a single purpose

Example

- Resource does not ensure required roles and features

- Responsibility of configuration to ensure prerequisites for resource

Process

Authoring

Write configuration in PowerShell

Compile to MOF

Staging

Clients receive configuration via push or pull

MOF is transferred

Make It So

Local Configuration Manager applies MOF using DSC resources

Style

Parameters

Validation

```
[Parameter(Mandatory)]  
[ValidateNotNullOrEmpty()]  
[string]
```

Parameter sets

```
[Parameter(ParameterSetName='Simple')]
```

Similar to PowerShell scripts

Input needs to be supplied explicitly

Configuration Data

No validation

Syntax is PowerShell Object Notation (PSON)
Semantics/validity must be checked separately

Similar to splatting

```
$Params = @{  
    ComputerName = 'dc-01.contoso.com'  
    Credential    = (Get-Credential)  
}  
New-PSSession @Params
```

Input can be pulled from entire ConfigData
Easy interface from third party systems

Complexity

Single Purpose

Configuration describes a single role
Service consists of separate configurations

Using parameters will not get you beyond this stage

Input required by roles must be supplied to separate configurations

- Duplicate information
- Hard to maintain

Many calls to generate MOFs for all involved nodes

Service Description

Configuration describes a whole service
All involved roles in a single bundle

Only ConfigData enables complex node configurations

Input is provided in a single data structure to the whole configuration

- Complexity remains in configuration
- Dependencies are resolved by configuration

Single call generates all MOFs

Attacks against Pull Servers

Man-in-the-Middle

Server authentication

Use HTTPS with trusted CA

Eavesdroppers

Protect GUID from harvesting

HTTPS helps as well

Foreign devices

Use certificate based client authentication

Known issues

[DSC LCM 2.0 Configuration does not work with Client Certificate Authentication](#)

[Do not enforce anonymous authentication for DSC Web Pull Server](#)

Node Configuration

Confidentiality

Node configuration contains crucial information

Use pull server with server and client authentication

Credential Encryption

Encrypt credentials and use minimal privileges

Known issues

Private Key not accessible for DSC LCM when key is generated using CNG instead of legacy CSP

GUID Management

Active Directory computer account GUID

Random GUID

GUID per computer or per service role

Node Configuration

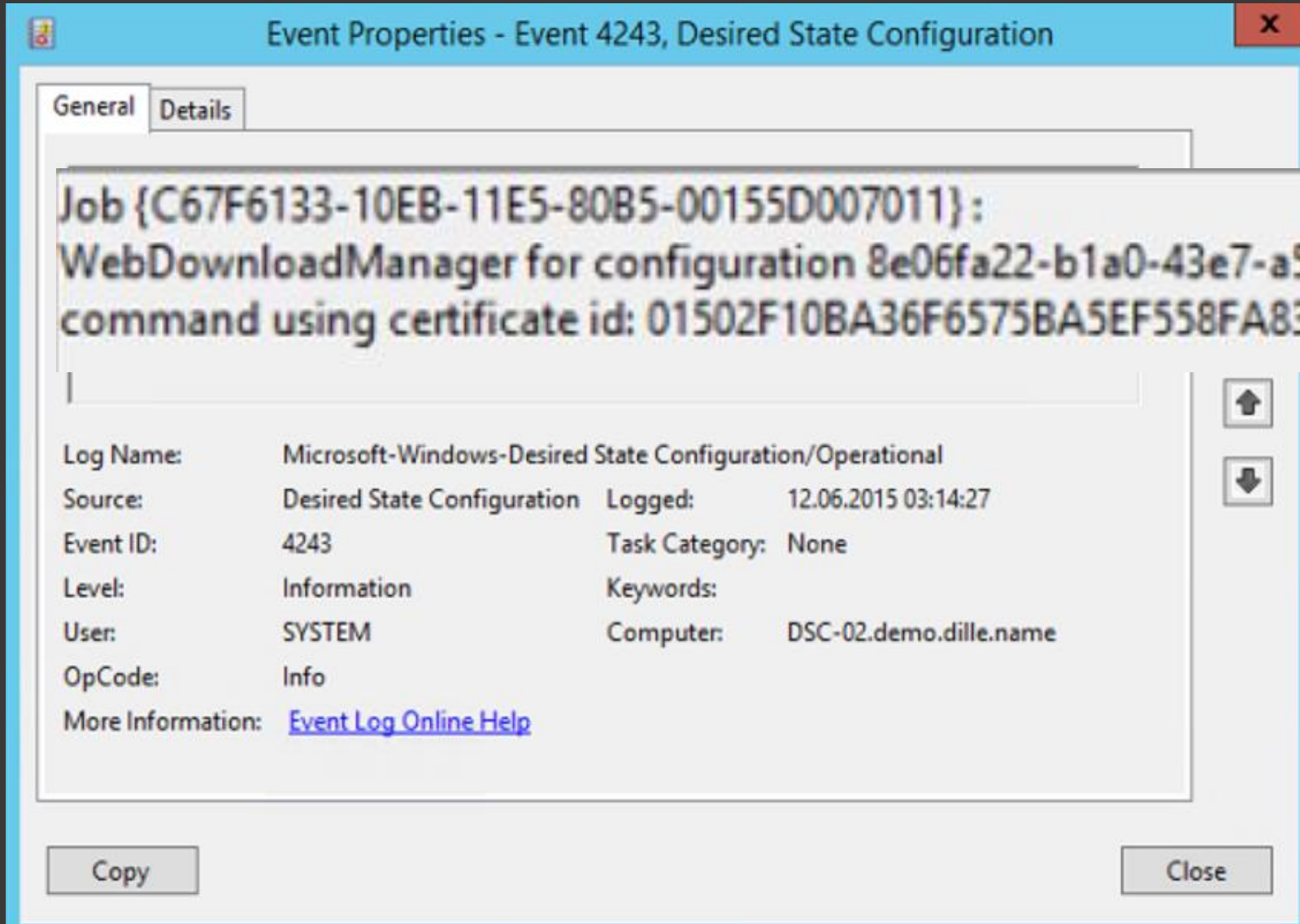
Example LCM Configuration

```
Configuration LCM {
  Node 'localhost' {
    LocalConfigurationManager {
      DownloadManagerCustomData = @{
        ServerUrl = 'https://$PullServer/PSDSCPullServer.svc',
        AllowUnsecureConnection = 'false',

        # Thumbprint of certificate for client authentication
        CertificateID = '01502f10ba36f6575ba5ef558fa8378d6fcaeb14'
      }
      # Thumbprint of certificate for credential encryption
      CertificateID = '01502f10ba36f6575ba5ef558fa8378d6fcaeb14'
    }
  }
}
```

See also [End-to-End Security](#) by [Ben Gelens](#)

Client Authentication



The screenshot shows a Windows Event Viewer window titled "Event Properties - Event 4243, Desired State Configuration". The "Details" tab is selected, displaying the following information:

Job {C67F6133-10EB-11E5-80B5-00155D007011}:
WebDownloadManager for configuration 8e06fa22-b1a0-43e7-a546-c7e4e94c2aae Do-DscAction command using certificate id: 01502F10BA36F6575BA5EF558FA8378D6FCAEB14.

Log Name:	Microsoft-Windows-Desired State Configuration/Operational		
Source:	Desired State Configuration	Logged:	12.06.2015 03:14:27
Event ID:	4243	Task Category:	None
Level:	Information	Keywords:	
User:	SYSTEM	Computer:	DSC-02.demo.dille.name
OpCode:	Info		
More Information:	Event Log Online Help		

Buttons: Copy, Close

Class-Based Resources

Requires WMF 5

Based on PowerShell classes

- Object-oriented design paradigm

Reduced overhead on authoring phase

- Parameters are defined once as properties instead of per function (Get, Test, Set)

Full integration with PowerShell modules

- No more directory called `DSCResources` inside module

Known Issues

Class defined DSC resource module cannot be acquired via Pull Server because of script module structure check of LCM WebDownloadManager

Remote Desktop Services

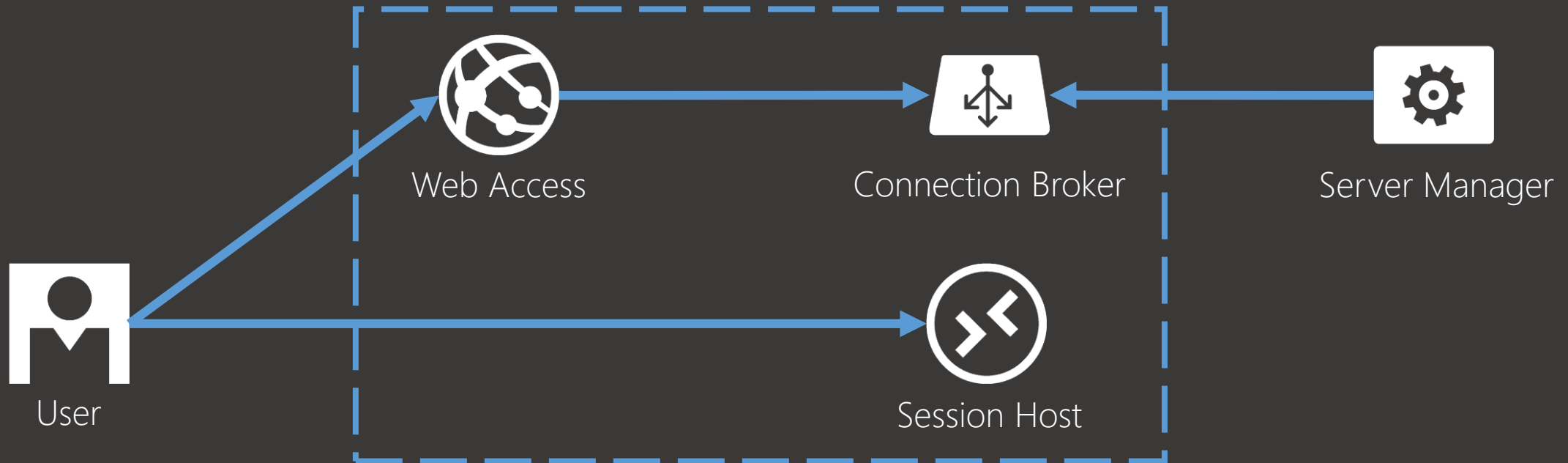
Short Introduction

User browses published resources in Web Access

Web Access contacts Connection Broker to determine appropriate Session Host

User connects directly to Session Host

Deployment is managed by Server Manager



cRemoteDesktopServices

cRDSessionDeployment

Creates a new session deployment

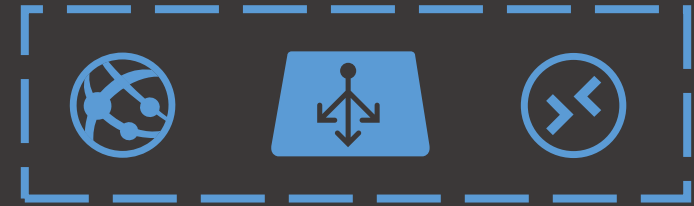
Supports quick and standard deployment

cRDSessionHost

Adds session host to existing deployment

cRDWebAccessHost

Adds web access to existing deployment



Available at GitHub

<https://github.com/nicholasdille/DSCResources>

Demo

Create new RD session deployment

[View Video](#)

Demo

Extend deployment with a new RD session host

[View Video](#)

Roadmap

vNext

cRDSessionCollection

- Creates a new session collection
- Configures a session collection

cRDRemoteApp

- Creates a new RemoteApp inside a collection
- Configures a RemoteApp

Future

cCBHA

- High availability for Connection Broker

cRDCertificate

- Manage certificates used in deployment (up to four)

cRDSHConfiguration

- Configure session hosts (e.g. licensing mode, listener security)

Caveats

Execution context

LCM makes it so as SYSTEM

Option 1: Invoke-Command

```
Invoke-Command -ComputerName $env:COMPUTERNAME -Credential $Cred -ScriptBlock {  
    <# ... #>  
}
```

Option 2: PsDscRunAsCredential (new in WMF 5 April Preview)

```
Configuration Test {  
    Node 'localhost' {  
        Script Test {  
            PsDscRunAsCredential = (Get-Credential)  
            GetScript = '@{}'  
            TestScript = '$false'  
            SetScript = {New-Item -ItemType File -Path \\server\share\file.txt}  
        }  
    }  
}
```

Caveats

Double-hop authentication

Remoting does not allow credential delegation to a third system by default

Solution: Use CredSSP

```
Invoke-Command -Comp $env:COMPUTERNAME -Auth Credssp -Cred $Cred -ScriptBlock {  
    <# ... #>  
}
```

CredSSP must be enabled

[Enable PowerShell Remoting with CredSSP using Group Policy](#)

PsDscRunAsCredential solves the double-hop problem

Separate credentials per resource

Now what?

Learn

Microsoft Virtual Academy (MVA)

[Getting Started with PowerShell Desired State Configuration \(DSC\)](#)

[Advanced PowerShell Desired State Configuration \(DSC\) and Custom Resources](#)

Book

[Windows Powershell Desired State Configuration Revealed](#) by Ravikanth Chaganti

My blog

[Tag: PSDSC](#)

[Useful Resources to Teach Yourself PowerShell DSC](#)

Get involved

GitHub

[PowerShell](#)

Stay up to date

Twitter

[#PSDSC](#)

#FF: [Steven Murawski](#), [Ravikanth Chaganti](#), [Michael Greene \(MSFT\)](#), [Ben Gelens](#), [Trevor Sullivan](#), [David O'Brien](#), [Fabien Dibot](#), [Jan Egil Ring](#), [Jacob Benson](#), [Narayanan Lakshmanan](#), [Nicholas Dille](#)

Web

[PowerShell Magazine](#)

[PowerShell.org](#)

Summary

Design Goals

Standards-Based
Declarative
Idempotent



Style

Configuration Data
Service Description



Security

HTTPS Pull
Cred Encryption
Client Auth



Custom Resources

cRemoteDesktopServices
Execution Context
Double-Hop Auth



Now what?

MVA
Book
GitHub
Twitter



Who is this guy?

Husband, father, geek, author, Aikidoka

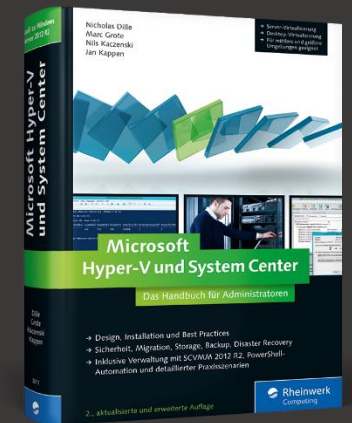
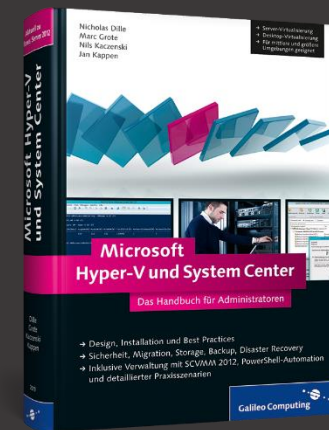
Teamlead Enterprise Consulting @ Makro Factory

10+ years of experience in SBC/VDI market

RDS MVP since 2010

<http://dille.name/blog>

@NicholasDille



Microsoft Hype-V und System Center
in Windows Server 2012 (R2)