Technical Report

Automating Microsoft Exchange Server 2010 Installation and Storage Provisioning—the Scripted Solution

Niyaz Mohamed, NetApp
August 2012 | TR-4085

TABLE OF CONTENTS

LIST OF FIGURES

1 Automating Microsoft Exchange 2010 Installation and Storage Provisioning Overview

Automating and virtualizing Exchange on NetApp® storage delivers significant benefits, including:

- Server and storage hardware cost reduction
- Space and power savings
- Improved server use
- Simplified management
- Repeatable, proven process to deploy the infrastructure
- Reduced human error resulting from fewer manual processes
- Advanced storage management, provisioning, backup, and data recovery features
- Shared virtual infrastructure that supports multiple platforms and applications

When you virtualize Exchange 2010 roles, NetApp recommends that these roles be separated onto different host servers. In the event of a host server failure, this separation prevents failure of any particular role. For example, deploying one Client Access Server (CAS), one Hub Transport Server (HUB), and two mailbox servers per host server provides a good mix in terms of distribution of roles.

For additional information and recommendations for virtualizing Exchange 2010 servers, refer to http://technet.microsoft.com/en-us/library/aa996719.aspx.

1.1 Purpose and Scope

Automation and virtualization of Exchange on NetApp storage provides significant reductions in cost and increases business agility. This report is intended to help reduce customer and partner deployment times by providing specific guidance for building on-premises Exchange environments that use NetApp storage on virtualized platforms.

1.2 Target Audience

NetApp recommends this report for the following audiences:

- Exchange architects
- Customer IT business leaders
- Private cloud architects
- NetApp field personnel and partners

1.3 Technology Solution

This packaged solution is a tightly integrated software management stack based on Microsoft® Windows® components and NetApp software components. Our goal is to provide an integrated management experience that allows customers to easily implement on-premises Exchange Server environments in both production and test environments.

1.4 Use Case Summary

Automation is a key theme among Exchange Server administrators, and the addition of the PowerShell™ components in Exchange 2010 enables administrators to automate almost every aspect of their daily operations.

This package is a multipurpose platform designed to accommodate Exchange Server workloads in an enterprise setting and also for lab environments. The key features that this platform offers are:

- **Easier deployment of Exchange Server 2010 SP2**
  Easier deployment is enabled by the use of PowerShell scripts that automate the Exchange Server 2010 SP2 installation. The current script provides the flexibility to install the prerequisites for Windows operating systems and also the hotfixes required for each role. Once the prerequisites are installed, the Exchange Schema version attribute is checked and, depending on the results, the schema is upgraded. The next step is the Exchange installation, which is accomplished using the installation scripts.
- **Storage provisioning**
  Storage provisioning is provided by the NetApp Data ONTAP® PowerShell Toolkit 2.0 cmdlets. This allows consistency of operations and permits the solutions team to provide operational procedures based on a consistent architecture. All operational procedures are developed through the use of these tools. Wherever possible, NetApp storage efficiency and data protection technologies are used as key capabilities of the storage offerings.
- **NetApp SnapDrive for Windows installation and configuration**
  SnapDrive® for Windows (SDW) installation is accomplished by using PowerShell and Windows batch files. This application helps with storage provisioning and managing disks in both physical and virtual environments. SDW manages the LUNs on the storage system, making them available as local disks on Windows hosts.
- **NetApp SnapManager for Exchange Server installation**
  NetApp SnapManager® for Exchange Server installation is also triggered using the PowerShell and Windows batch files. SnapManager for Exchange (SME) provides an integrated data and storage management solution for Microsoft Exchange Server 2010 that enhances the availability, scalability, and reliability of Exchange databases. SME provides rapid online backup and restoration of databases, along with local or remote backup set mirroring for disaster recovery.

  SME uses online Snapshot™ technologies that are part of Data ONTAP to integrate with Exchange backup and to restore APIs and the Volume Shadow Copy Service (VSS). SME also uses SnapMirror® technology to support disaster recovery for Exchange 2010 environments that are not using Database Availability Group (DAG) designs for disaster recovery or for non-DAG environments.

  SME provides the following data management capabilities:

  - Migrating Exchange databases and transaction logs to NetApp LUNs
  - Backing up Exchange databases and transaction logs from NetApp LUNs
  - Verifying Exchange databases and transaction logs in backup sets
  - Managing backup sets
  - Archiving backup sets
  - Restoring Exchange databases and transaction logs from previously created backup sets
- **NetApp Single Mailbox Recovery (SMBR)**

  This mailbox recovery software that saves time, money, and resources when recovering Microsoft Exchange data can also be automated using PowerShell. The NetApp Single Mailbox Recovery tool allows the customer to extract e-mails and other items from an Exchange database and place them into a PST file or a live mailbox. In order for you to extract e-mail from an Exchange database, the LUN must be mounted using SnapDrive for Windows.

2 Architecture Primary Use Case

NetApp solutions are designed to be tightly integrated with Microsoft Exchange Server 2010. Running Microsoft Exchange 2010 in a virtualized environment with NetApp storage enables better availability and flexibility and more efficient use of infrastructure with no impact on performance. By using an automated and virtualized environment, IT organizations can make much more effective use of their server installations. In certain scenarios these organizations can limit the number of servers being used for

Exchange by using virtual machines instead of the physical servers that would otherwise be needed to support Exchange.

The combination of virtualization and NetApp solutions provides an architectural design that facilitates consolidation in the following ways.

- Virtualization platforms can simultaneously support VMs running different operating systems on the same physical server.
- All NetApp systems, from entry level to high end, are based on a single unified storage architecture that simplifies management with common tools and processes for both physical and virtual environments. A single NetApp system supports major storage protocols, including Fibre Channel and iSCSI, and both Fibre Channel and SATA disk drives; it can be used to consolidate all data types.

Note: This package uses the iSCSI protocol.

By using NetApp as the storage infrastructure, the virtualized environment benefits from the advanced storage management, provisioning, backup, and data recovery features of NetApp as well as significantly greater storage efficiency. NetApp allows you to add storage as needed by running under a hypervisor platform for added efficiency. You can also quickly provision additional Exchange servers including the HUB transport, Client Access Server, and Mailbox server roles.

The virtualization platform, through its ability to virtualize Exchange, assists in efficient use of hardware resources that can be combined with the other key advantages of server virtualization, which include better availability, lower cost, and increased flexibility. You can realize multiple benefits from using Exchange in a virtualized environment with NetApp storage technology, including:

- **Effective use of server hardware.** Migrating the entire Exchange environment from dedicated physical servers that have relatively low utilization rates can lead to significantly higher server utilization.
- **Consolidation.** Small and medium-sized organizations, as well as remote offices for larger organizations, can combine Exchange Client Access Server and HUB Transport Server roles into a virtualized platform with other application servers on the same physical server.
- **Savings.** Save on power and space.
- **Reduced server hardware requirements.** The number of physical servers required to support Exchange can be reduced by adding multiple Exchange roles to a single virtual machine.

IT administrators overwhelmingly agree that it makes sense to automate their Exchange installations. It provides a solid, repeatable process by which to deploy the infrastructure, and it takes less time to complete the process. By reducing human error, organizations that automate Exchange deployment and configuration can improve the overall quality of their infrastructure.

2.1 Server

The host server architecture is a critical component of the virtualized infrastructure as well as a key variable in the consolidation ratio and cost analysis.

The system architecture of the host server refers to the general category of the server hardware itself. The primary factor to consider when selecting a system architecture is that each virtualization host can contain multiple guests with multiple workloads. The critical factors are processor, RAM, storage, network capacity, high I/O capacity, and low latency. The host server must be able to provide the required capacity in each of these categories.

2.2 Storage

The storage design for any virtualization-based solution is a critical element that typically accounts for a large percentage of the solution’s overall cost, performance, and agility.

Although many storage options exist, organizations should choose their storage devices based on their specific data management needs. Storage devices typically include modular, flexible midrange SANs and high-end SANs. Modular midrange SANs are procured independently and can be chained together to provide greater capacity. NetApp FAS series controllers are efficient, can grow with the environment as needed, and require less up-front investment than high-end SANs.

Large enterprises might have larger storage demands and might need to serve a larger set of customers and workloads. In this case, high-end SANs can provide the highest performance and capacity. High-end SANs typically include more advanced features such as continuous data availability through technologies like replication and clustering. The NetApp FAS series unified architecture offers all of these capabilities—including replication and clustering support for all FAS controllers—from the smallest 2000 series to the largest 6300 series controller. The FAS controllers used in this solution should be licensed for the full range of NetApp Data ONTAP features.

The features of Microsoft Exchange Server offer great advantages only if the underlying storage can support their availability and performance requirements under all operational conditions. NetApp iSCSI solutions for Microsoft Exchange Server 2010 deliver superior performance and advanced features such as online Snapshot copies and the ability to reconfigure storage with no downtime.

2.3 Virtualization

Virtualization is based on the abstraction of physical system resources so that multiple logical partitions can be created and can host a wide range of operating systems that run simultaneously on a single physical server. Rather than paying for many underutilized servers, each dedicated to a specific workload, server virtualization allows those workloads to be consolidated onto a smaller number of more efficiently utilized physical systems. Server virtualization provides the following benefits.

- Consolidates multiple underutilized physical servers on a single host running VMs
- Reduces workforce, space, and kilowatts used by taking advantage of virtualization for server and storage consolidation and agility

Virtualization can also help to simplify and accelerate provisioning of virtual machines. The vast majority of virtualized Exchange 2010 architectures are deployed on Hyper-V™ and VMware® virtualization platforms. Please refer to Microsoft TechNet article Understanding Exchange 2010 Virtualization during the planning phase before using this package for production or test environments.

3 Storage Efficiency

In simple terms, storage efficiency means increasing storage utilization and decreasing storage costs. This guide implements the following technologies to enhance storage efficiency and to optimize the existing storage in the infrastructure while deferring or avoiding future storage needs.

RAID-DP. RAID-DP® technology is NetApp's implementation of double-parity RAID 6, which is an extension of the original NetApp Data ONTAP WAFL® RAID 4 design. Unlike other RAID technologies, RAID-DP provides the ability to achieve a higher level of data protection without any performance effect while consuming a minimal amount of storage.

SATA. The performance acceleration provided by WAFL and the double-disk protection provided by RAID-DP make economical, large-capacity SATA drives practical for production application use. In addition, to negate the read latencies associated with large drives, SATA drives can be used with the NetApp Flash Cache card, which significantly increases performance with large working set sizes.

Snapshot capability. NetApp Snapshot technology provides zero-cost, near-instantaneous backup, point-in-time copies of the volume or LUN by preserving Data ONTAP WAFL consistency points (CPs). Creating Snapshot copies incurs minimal performance effect because data is not moved.

NetApp deduplication. The deduplication process stores only unique blocks of data in the volume and creates additional metadata.

The core enabling technology of deduplication is fingerprints. When deduplication runs for the first time on a FlexVol® volume, it scans the blocks and creates a fingerprint database that contains a sorted list of all fingerprints for used blocks in the flexible volume. Each 4KB block in the storage system has a digital fingerprint, which is compared to other fingerprints on the volume. If two fingerprints are found to be the same, a byte-for-byte comparison is done of all bytes in the block. If they are an exact match, the duplicate block is discarded and the space is reclaimed.

Deduplication consumes system resources and can alter the data layout on disk. Due to the application's I/O pattern and the effect of deduplication on the data layout, read and write I/O performance can vary.

Note: Deduplication is transparent to Exchange, and the block changes are not recognized by Exchange. Therefore, the Exchange database remains unchanged in size from the host's perspective, even though capacity savings occur at the volume level.

For best practices on how to configure Microsoft Exchange Server 2010 for Data ONTAP 7-Mode storage systems, refer to TR-4033: Microsoft Exchange Server 2010 and SnapManager for Exchange Best Practices Guide.

4 Data Protection

4.1 Backup and Disaster Recovery

This solution provides a means of data backup and disaster recovery. The combination of NetApp SnapManager products allows the solution to be backed up by using NetApp native Snapshot technology.

SnapManager for Exchange is tightly integrated with Microsoft Exchange, which allows for consistent online backups of your Exchange environment while leveraging NetApp Snapshot copy technology. SnapManager for Exchange is a Volume Shadow Service (VSS) (Snapshot copy) requestor, which means that it uses the Microsoft VSS subsystem to initiate backups.

NetApp SnapManager for Microsoft Exchange is an integral component of the NetApp data management solution for Microsoft Exchange Server environments. By reducing backup and restore times, minimizing Exchange outages, and consolidating Exchange storage, SnapManager for Exchange delivers a cost-effective solution for managing critical Exchange data.

5 Storage Layout Planning

5.1 Aggregate Recommendations

This automation script applies NetApp best practices for Microsoft Exchange Server 2010 with respect to aggregate planning. In this solution, a new and separate aggregate is created for the Exchange workload.

Pooling all of the available disks into a single, large aggregate might maximize performance; however, it might not meet the data availability requirements set forth in the service-level agreement (SLA). Creating separate aggregates for Exchange database volumes and Exchange transaction log or SnapInfo volumes can meet the performance requirements of Exchange Server 2010 while providing the data availability required by most typical SLAs. In the unlikely event that an aggregate is lost, part of the Exchange data is still available. An Exchange administrator can potentially recover data from the available aggregate.

For the best practices on configuring aggregates for Microsoft Exchange Server 2010 for 7-Mode storage systems, refer to TR-4033: Microsoft Exchange Server 2010 and SnapManager for Exchange Best Practices Guide.

5.2 Volume Planning

In this solution, separate volumes are created for each database and for transaction logs.

Data ONTAP enables the creation of flexible (FlexVol) volumes for managing data without the need to assign physical disks to the volumes. Instead, the FlexVol volumes enjoy performance benefits from a larger pool of physical disks called an *aggregate*. This results in the following additional benefits for Microsoft Exchange environments.

- A large number of volumes can be created, all with independent Snapshot copy schedules, mirroring policies, and so on.
- All volumes can be managed independently while receiving the maximum I/O benefit of a much larger pool of disks.

Volume layout is critical in creating and sustaining a highly available Exchange environment. Careful consideration of various backup groups, disaster recovery scenarios, and even archiving solutions helps determine the placement of volumes onto aggregates and the placement of the corresponding LUNs onto those volumes.

For best practices on planning volumes for Microsoft Exchange Server 2010 for 7-Mode storage systems, refer to TR-4033: Microsoft Exchange Server 2010 and SnapManager for Exchange Best Practices Guide.

5.3 LUN Layout Recommendations

When considering an Exchange 2010 LUN configuration, the number of LUNs you should provision largely depends on your recovery point objectives (RPOs) and your recovery time objectives (RTOs). This guide considers the best practices for LUN layout in the context of SnapManager for Exchange and creates a separate volume and LUN for databases and logs.

Two LUNs per Database and One LUN per Volume

In this solution, each mailbox database and transaction log set is placed on a separate LUN. This solution provides greater flexibility in terms of recovery options but increases the total number of LUNs required.

For the best practices for creating LUNs for Microsoft Exchange Server 2010 for 7-Mode storage systems, refer to TR-4033: Microsoft Exchange Server 2010 and SnapManager for Exchange Best Practices Guide.

6 Capacity Planning

A properly sized Exchange environment meets both the Microsoft requirements for Exchange storage and the customer's requirements as indicated in the SLAs. To help obtain a properly sized environment, tools convert information about the customer environment into a physical storage recommendation. Use the following primary tools when planning an Exchange environment for a customer.

- Microsoft storage calculator.
- NetApp Exchange Sizing Tool; please work with your local NetApp partner or your NetApp representative to enable proper sizing.

The sizing information provided by these tools is an important component of planning an Exchange environment and provides a framework for storage group layout and LUN requirements. Keep in mind that the Microsoft storage calculator does not make recommendations on storage design (RAID parity, number of disks, and so on), because the storage design largely depends on the type of storage array being used. When sizing Exchange Server deployments using NetApp storage, it is important to use the NetApp System Performance Modeler with the data from the Microsoft Exchange 2010 Mailbox Server Role Requirements Calculator.

Best Practice
Use the NetApp System Performance Modeler for Exchange to size all Exchange Server deployments that use NetApp storage. Consult a local NetApp Exchange expert or your NetApp partner to assist in accurately sizing Exchange Server 2010.

Note: Using the previously mentioned tools, size the Exchange environment before using the scripts to deploy the Exchange Servers and provision storage.

7 Storage Automation

One of the objectives of automating and virtualizing an Exchange Server is to enable automation of the Exchange Server deployment in a correct and effective manner. Doing this on a large scale requires tight integration with the storage architecture as well as robust automation.

NetApp supports a wide range of Microsoft management tools and APIs. Specifically, NetApp has shipped the Data ONTAP PowerShell Toolkit, which allows the management of NetApp controllers from PowerShell.

The automation in this document uses NetApp Data ONTAP PowerShell Toolkit cmdlets and Windows PowerShell cmdlets in PowerShell scripts.

7.1 Considerations

Infrastructure Deployment

When deploying several Exchange servers, choosing the right deployment model is critical. Choices range from manual installation, which is highly inefficient, through varying degrees of automation up to enterprise-class management systems. To achieve the architecture principle of predictability, all infrastructure components should be able to be deployed and configured in a repeatable and automated fashion. Examples include configuring Exchange infrastructure and storage infrastructure and provisioning servers.

The key components of a successful deployment are the standard Windows Server® roles such as Active Directory® Domain Services and Domain Name System (DNS). Using these technologies, it is possible to provide a robust deployment infrastructure using standard in-box solutions and toolkits.

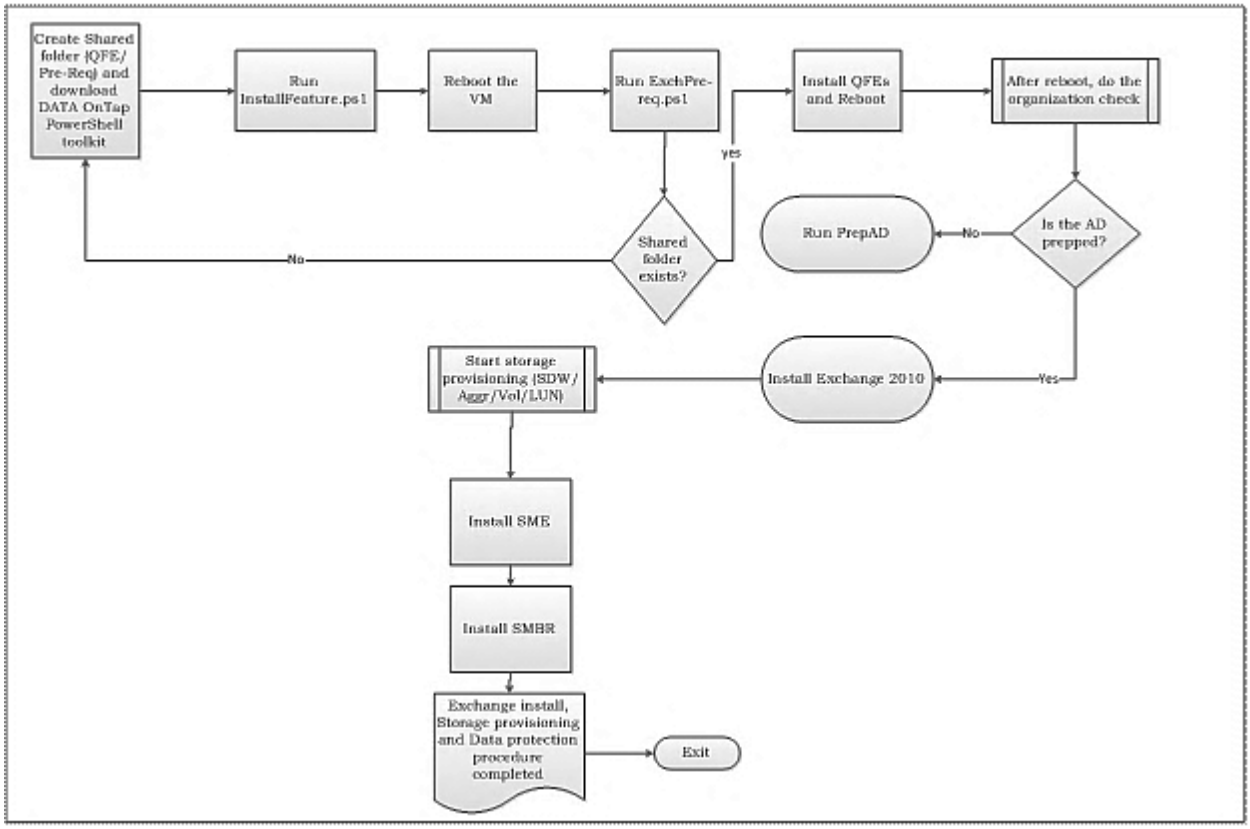
The process and automation detailed in this document require that you:

- Make sure the virtual machines are sized and provisioned specifically for the Exchange role to be installed.
- Install Windows Server 2008 R2 Enterprise/Datacenter Edition.
- Confirm that the NetApp Data ONTAP PowerShell Toolkit is installed on any host where provisioning tasks are performed. You can download the toolkit here.
- This report assumes that the account you use to execute the scripts for installing Exchange Server 2010 has been delegated the appropriate administrative and management rights.

8 Automating Exchange Server Deployment

Figure 1 illustrates the workflow for the Exchange deployment process.

Figure 1) Exchange deployment process.



8.1 The Workflow

Before you begin, verify that the virtualization environment is properly set up and the respective virtual machines are domain joined.

The steps shown in Figure 1 represent the sequence of steps involved in Microsoft Exchange Server 2010 deployment and storage provisioning.

Note: The virtual machines should be built and configured according to the Microsoft recommendations before you use this package.

1. Download the required hotfixes and software for executing the scripts.
2. Install the Data ONTAP PowerShell Toolkit on the designated virtual machine.
3. Create three folder structures on a designated CIFS share to copy the mandatory software, hotfixes, and setup files that were downloaded in step 1.
4. Copy the software such as Filterpack, SnapDrive for Windows, SnapManager for Exchange, and Single Mailbox Recovery to the root of the “Pre-req” folder so that it can be used as the installation directory for software installation.
5. Copy the required hotfixes such as KB977020, KB979744, KB982867, KB983440, and KB979099 to the “QFE” folder under the Pre-req folder so that the relevant hotfixes can be installed using the scripts.
6. Copy the Exchange setup files to the “ExchangeMedia” folder on the designated CIFS share.
7. Install the mandatory prerequisites and reboot the server.
8. Run PrepareAD, which in turn runs all of its relevant tasks. PrepareAD also executes PrepareSchema, which updates the schema to accommodate Exchange Server 2010.

9. Install the required Exchange 2010 roles according to the requirement.
10. Install SnapDrive for Windows.
11. Provision the storage by creating the aggregate, volume, and LUNs according to the design output.
12. Clone the volume and LUNs according to the design requirement provided during volume and LUN creation.
13. Install SME and manually configure it by using the configuration wizard.
14. Install SMBR to use the “Run SMBR” feature in SME. Make sure 32-bit Outlook is installed on the server before installing the SMBR software.

8.2 The Scripts

Eight scripts contain the various functions that are used for Exchange automation and storage provisioning and SME installation. Verify that the Data ONTAP PowerShell Toolkit is installed on the Exchange Server where you plan to provision the storage, including the iSCSI LUNs.

The scripts are as follows:

- Logger.ps1
- InstallOSFeatures.ps1
- ExchPrereq.ps1
- PrepareAD.ps1
- InstallExchroles.ps1
- Map_luns.ps1
- SMEinstall.ps1
- SMBRinstall.ps1

Prerequisites

Take the following actions before running the scripts.

1. Provide the necessary parameters that are required by the scripts.
2. Download the necessary hotfixes and software for deploying Exchange and provisioning storage.
3. Install the Data ONTAP PowerShell Toolkit on the designated server where the provisioning operation occurs.
4. Create the mandatory ExchangeMedia and Pre-req folders, copy the relevant files downloaded in Step 2 to the respective folders, and confirm the share path access.
5. Confirm that the execution policy for scripts is set to “RemoteSigned.”
6. Use PowerShell or PowerShell ISE to execute the scripts accordingly.

The following subsections elaborate on the scripts mentioned in this section.

InstallOSFeatures.ps1

This script automates the process of installing the prerequisites for the Windows OS. The script uses the parameter block and provides the options to key in the roles according to the projected design.

ExchPrereq.ps1

This script triggers installation of the hotfixes required for Exchange Server 2010 and also configures Net.Tcp Port Sharing Service for automatic startup. A system restart is required.

Note: If you are installing Exchange 2010 SP2 on a server running Windows 2008 R2 SP1, you are not required to install these hotfixes separately.

The following hotfixes are required for the Mailbox, Client Access, and Hub Transport servers for Windows Server 2008 R2 RTM.

- Knowledge Base article 979099, *An update is available to remove the application manifest expiry feature from AD RMS clients*. Without this update the AD RMS features might stop working.
- Knowledge Base article 979744, *A .NET Framework 2.0-based Multi-AppDomain application stops responding when you run the application*.
- Knowledge Base article 983440, *An ASP.NET 2.0 hotfix rollup package is available for Windows 7 and for Windows Server 2008 R2*.
- Knowledge Base article 977020, *FIX: An application that is based on the Microsoft .NET Framework 2.0 Service Pack 2 and that invokes a Web service call asynchronously throws an exception on a computer that is running Windows 7*.
- Knowledge Base article 982867, *WCF services that are hosted by computers together with a NLB fail in .NET Framework 3.5 SP1*.

PrepareAD.ps1

The script calls the PrepareAD cmdlet. The script also checks for an existing organization name or else prompts to provide the organization name as part of the parameter declaration.

For more information, refer to Microsoft TechNet http://technet.microsoft.com/en-us/library/bb125224.aspx.

InstallExchroles.ps1

The `Install.PS1` script is used to install the different roles (Mailbox, Client, and Hub) according to the client or customer requirements. The process includes copying the Exchange setup files to a designated folder on the Exchange Server and later uses the unattended installation capability of Exchange to install the relevant roles.

Map_luns.ps1

This script is used to install SDW and to configure it using the relevant settings.

Once SDW is installed, the script continues by creating the aggregates, volumes, and iSCSI LUNs, depending on the number and size specified by the customer. Then it clones the volumes and LUNs, provided the required licenses are installed on the NetApp storage controllers.

SMEinstall.ps1

This script installs SME in silent mode; however, the configuration must be done manually, which requires use of the SME GUI (configuration wizard).

Additional PowerShell cmdlets can be used to move the database and logs to the NetApp LUNs for the smooth functioning of SME.

SMBRinstall.ps1

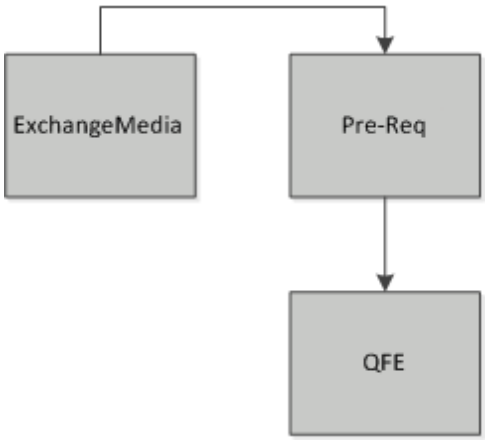
This script installs Single Mailbox Recovery in the silent mode using `msiexec`; however, you must first install Outlook to enable the functioning of SMBR, and the SMBR Administrative Server should be configured separately according to the Single Mailbox Recovery Administrative Server User Guide.

8.3 Mandatory Folder Structure

This process is formulated with an assumption that the required folders are created before the script for the deployment procedure is executed. The required hotfixes and Exchange setup files should be copied

to the respective folders in a CIFS share during the initial phase so that scripts can complete seamlessly. The folder structure is shown in Figure 2.

Figure 2) Required folder structure.



- **ExchangeMedia Folder.** This folder holds the Exchange 2010 setup files so that it can be copied over to the temp folder on the dedicated servers.
- **Pre-Req Folder.** This folder is the repository for all required software that is needed for using the script correctly.
- **QFE Folder.** The QFE folder is the location for keeping all the required hotfixes for the Exchange Server 2010 implementation.

8.4 Logging

All of these scripts also provide a mechanism for generating a centralized report file that can be used to identify success or failure of the scripts, and the log file can be found at `%systemdrive%\NVAlogs`.

9 Conclusion

This guide was created to help customers and partners with automation and virtualization of Exchange deployments. Each specific implementation combines the virtualization platform, Microsoft software, consolidated guidance, and validated configurations with NetApp technology—including deployment and storage architectures and value-added software components. By utilizing this guide, organizations can rapidly deploy the Exchange on-premises environment while reducing their risk and total cost of ownership. The ultimate goal is to give organizations both the control and flexibility to automate and virtualize their Exchange servers.

Appendixes

Supporting Documents

The following references were used in this technical report:

- WP-7132: NetApp Hyper-V Cloud Fast Track with Cisco: www.netapp.com/us/library/white-papers/wp-7132.html
- RA-0008: Microsoft Private Clouds and NetApp Reference Architecture: http://media.netapp.com/documents/rp-microsoft-private-clouds.pdf

- TR-4033: Microsoft Exchange Server 2010 and SnapManager for Exchange Best Practices Guide: http://media.netapp.com/documents/tr-4033.pdf
- TR-4056: Microsoft Exchange Server 2010 and SnapManager for Exchange on Data ONTAP 8.1 Cluster-Mode Best Practices Guide http://media.netapp.com/documents/tr-4056.pdf
- NetApp SnapDrive for Windows: https://support.netapp.com/documentation/productlibrary/index.html?productID=30049
- NetApp SnapManager for Microsoft Exchange Server: http://support.netapp.com/documentation/productlibrary/index.html?productID=30034
- TR-3701: NetApp and Microsoft Virtualization: Solution and Implementation Guide http://media.netapp.com/documents/tr-3701.pdf
- TR-3702: NetApp Storage Best Practices for Microsoft Virtualization and NetApp SnapManager for Hyper-V http://media.netapp.com/documents/tr-3702.pdf

Scripts

InstallOSfeatures.ps1 (OS Feature Install)

```
param
(
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="Exchange Role to be installed")]
[String[]]
[ValidateNotNull()]
[ValidateSet("Hub Transport", "Client Access Server", "Mailbox", "Unified Messaging", "Edge
Transport")]
$ExchRole
)
Set-Variable -Name feature -Value $ExchRole -Scope global
Import-Module .\logger.ps1
Logger "--------------------------------------------------------------------------------"
Logger "Running Installfeatures script..."

Import-Module ServerManager

## feature installation
if ($feature -contains "Edge Transport") {
    Logger "Installing roles for Edge Transport"
    Add-WindowsFeature NET-Framework,RSAT-ADDS,ADLDS
} else {
    Logger "Installing features.."
    $installfeatures = "NET-Framework", "RSAT-ADDS", "Web-Server", "Web-Basic-Auth", "Web-
Windows-Auth", "Web-Metabase", "Web-Net-Ext", "Web-Lgcy-Mgmt-Console", "WAS-Process-Model",
"RSAT-Web-Server"
    if ($feature -contains "Client Access Server") {
        Logger "Installing roles for Client Access Server"
        $CASroles = "Web-ISAPI-Ext", "Web-Digest-Auth", "Web-Dyn-Compression", "Web-WMI", "Web-
Asp-Net", "Web-ISAPI-Filter", "Web-Client-Auth", "Web-Dir-Browsing", "Web-HTTP-Errors", "Web-
HTTP-Logging", "Web-HTTP-Redirect", "Web-HTTP-Tracing", "Web-Request-Monitor", "Web-Static-
Content","NET-HTTP-Activation","RPC-Over-HTTP-Proxy"
        $installfeatures = $installfeatures + $CASroles
    }
    if ($feature -contains "Unified Messaging") {
        Logger "Installing roles for UM"
        $UMroles = "Desktop-Experience"
        $installfeatures = $installfeatures + $UMroles
    }
    Add-WindowsFeature $installfeatures -Restart
}
```

ExchPrereq.ps1 (Exchange Prerequisites Install)

```
Import-Module .\logger.ps1
Logger "--------------------------------------------------------------------------"
Logger "Running Exchange Server 2010 Prerequisites script"

# Verifying the whether the OS version is Windows 2008 R2 and 64 bit
if (-not((Get-WMIObject win32_OperatingSystem).OSArchitecture -eq '64-bit') -and (Get-WMIObject
win32_OperatingSystem).Version -eq '6.1.7600'){
        Logger "Exchange Server 2010 should only be installed on 64 bit Windows 2008, Please use
the correct and supported OS version"
        Exit
}
Function global:GetInput
{
param
(
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
[String]
$Path
,
[parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]

[System.Management.Automation.PSCredential]
$Credential
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="Please mention the role to be installed. ")]
[String[]]
[ValidateNotNull()]
[ValidateSet("Hub Transport", "Client Access Server", "Mailbox", "Unified Messaging", "Edge
Transport")]
$Role
)

Set-Variable -Name credential -Value $Credential -Scope global
Set-Variable -Name share_path -Value $Path -Scope global
Set-Variable -Name feature -Value $Role -Scope global

$nwcred_username = $credential.getnetworkcredential().UserName
$nwcred_password = $credential.getnetworkcredential().Password
net use $share_path $nwcred_password /USER:$nwcred_username
if (Test-Path $share_path) {
    Logger "$path is validated with $($credential.UserName) and Role to be installed is $Role"
} else {
    Logger "Cannot connect to $path. Please check the Path and Credentials!"
    exit
}

$QFE_path = "$share_path\Pre-Req\QFE"
$ExchMediapath = "$share_path\ExchangeMedia"

Set-Variable -Name QFE_path -Value $QFE_path -Scope global
Set-Variable -Name ExchMediapath -Value $ExchMediapath -Scope global

Logger "Please copy all the required QFEs to $share_path\Pre-Req\QFE."
Logger "Please copy the Exchange Media contents to $share_path\ExchangeMedia."
}

Function InstallFilterPack(){
#Verify if Filter pack is already installed using the registry  key
HKLM:\Software\Microsoft\CurrentVersion\Uninstall\{95120000-2000-0409-1000-0000000FF1CE}

  $filterpackinstall = $true
```

ExchPrereq.ps1 (Exchange Prerequisites Install)

```
  $UninstallKey="SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall"
  $reg=[microsoft.win32.registrykey]::OpenRemoteBaseKey('LocalMachine','.')
  $regkey=$reg.OpenSubKey($UninstallKey)
  $subkeys=$regkey.GetSubKeyNames()
  foreach ($subkey in $subkeys) {
    if ($subkey -eq "{95140000-2000-0409-1000-0000000FF1CE}") {
        Logger "Filterpack installed"
        $filterpackinstall = $false
        break
    }
  }
    if ($filterpackinstall -eq $false) {return}
       $folderPath = 'C:\Temp'

       #Check if Temp folder exists, if not, create it
       if (Test-Path $folderpath){
              Logger "The folder $folderPath exists."
       } else{
              Logger "The folder $folderPath does not exist, Creating one"
              New-Item $folderpath -type directory
              Logger "Created $folderpath!"
       }

       # Check if the file exists, if not, copy it from the file share created as part of the
Pre-requisites
       $file = $folderPath+"\FilterPack64bit.exe"
       if (Test-Path $file){
              Logger "The file $file exists."
       } else {
              Logger "Copying from $share_path\Pre-Req"
        Copy-Item "$share_path\Pre-Req\FilterPack64bit.exe" $folderpath
              Logger "copied FilterPack!"
       }
       #Install Microsoft Filter Pack
       Logger "Installing Microsoft Filter Pack..."
       $pack = "cmd.exe /c "+$folderPath+"\FilterPack64bit.exe /quiet /norestart"
       Invoke-Expression $pack
       Start-Sleep -Seconds 20
       Logger "Installed FilterPack!"
}

Function InstallQFE {

    param([string]$role)
    if ($role -ieq "HUB") { $KBList = "KB982867","KB977020","KB983440","KB979744" }
    elseif ($role -ieq "CAS") { $KBList = "KB982867","KB977020","KB983440","KB979744" }
    elseif ($role -ieq "MBX") { $KBList = "KB982867", "KB979099" }
    else { Logger "$role is Invalid" }

    foreach ($KB in $KBList) {
    #$chkforQFE = systeminfo | findstr.exe $KB
    #if ($chkforQFE -eq $null) {
      if (!(Test-Path -Path "$QFE_path\*$KB*.msu")) { Logger "One of the required QFEs is
missing in the $QFE_path!! please refer to the documentation for a list of required QFEs,
download and copy them to $QFE_path and run the script again!" }
      $KB_file = (Get-ChildItem -Path $QFE_path\`*$KB`*.msu).name
      $qfe_install = "cmd /c $QFE_path\$KB_file /quiet /norestart"
      Invoke-Expression $qfe_install
      Logger "Installed $KB"
      #}
    }
    #Restart-Computer
}


Function SetRunOnce(){
       $hostname = hostname
              $RunOnceCommand = "sc \\$hostname config NetTcpPortSharing start= auto"
```

ExchPrereq.ps1 (Exchange Prerequisites Install)

```
        if (Get-ItemProperty -Name "NetTCPPortSharing" -path
'HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce' -ErrorAction SilentlyContinue) {
           Logger "Registry key
HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce\NetTCPPortSharing already exists."
       Set-ItemProperty "HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce" -Name
"NetTCPPortSharing" -Value $RunOnceCommand | Out-Null
         } else {
           New-ItemProperty "HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce" -Name
"NetTCPPortSharing" -Value $RunOnceCommand -PropertyType "String" | Out-Null
         }

}
GetInput

foreach ($item in $feature) {
    if ($item -ieq "Hub Transport") {
            InstallFilterPack
            SetRunOnce
            InstallQFE "HUB"
    } elseif ($item -ieq "Client Access Server") {
            SetRunOnce
            InstallQFE "CAS"
    } elseif ($item -ieq "Mailbox") {
            InstallFilterPack
            SetRunOnce
            InstallQFE "MBX"
    } elseif ($item -ieq "Unified Messaging") {
            SetRunOnce
            InstallQFE "UM"
    } else
    {
            Logger "Not a valid Role to install"
    }
   Logger "Installed Pre-requisites necessary for Exchange Server installation"
}
Restart-Computer -Force
```

PrepareAD.ps1 (Prepare Schema and AD)

```
# Prepares AD for Exchange
function PrepareAD
{
    param
    (
    [parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
    [String]
    $Path
    ,
    [parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]
    [System.Management.Automation.PSCredential]
    $Credential
    ,
    [parameter(Mandatory=$true,
           HelpMessage="organization name")]
    [string]
    [AllowNull()]
    [AllowEmptyString()]
    $Organization
    )

    Set-Variable -Name credential -Value $Credential -Scope global
    Set-Variable -Name share_path -Value $Path -Scope global
```

PrepareAD.ps1 (Prepare Schema and AD)

```
    Set-Variable -Name Organization -Value $Organization -Scope global
    Import-Module .\logger.ps1
    Logger "--------------------------------------------------------------------------------"
    Logger "Running PrepADtest..."

    $nwcred_username = $credential.getnetworkcredential().UserName
    $nwcred_password = $credential.getnetworkcredential().Password
    net use $share_path $nwcred_password /USER:$nwcred_username
    if (Test-Path $share_path)
    {
        Logger "$path is validated with $($credential.UserName) and Role to be installed is
$Role"
    }
    else
    {
        Logger "cannot connect to $path. Please check the path and credentials!"
        exit
    }
    #Check for orgname by creating a new directory object to query configuration information in
Active Directory so we can query AD to retrieve what we need
        $configNC=([ADSI]"LDAP://RootDse").configurationNamingContext
        $ADobj = "LDAP://" + $configNC
        $ADobjquery = [ADSI]$ADobj

    # Creating a directory search object and Filtering with the class of the Exchange Org
Container
        $findobj = [System.DirectoryServices.DirectorySearcher]$ADobjquery
        $findobj.Filter = "(objectclass=msExchOrganizationContainer)"
        #$findobj.SearchRoot = $ADobjquery
        $ObjContainer = $findobj.Findone()

    if (!($ObjContainer))
    {
        if (!($Organization))
        {
            Logger "Please provide organisation name with the -Organization switch for first
time installation"
            exit
        }
        else
        {
            $Org_name = $Organization
        }
    }
    else
    {
        $Org_name = ((((($ObjContainer.GetDirectoryEntry().distinguishedname) -split (","))[0])
-split ("="))[1]
    }


        # Copy the Exchange Setup files to the designated server.
        Logger "Copying the Exchange Setup files to the designated server so as to execute
PrepareAD"
        Copy-item -Path "$share_path\Exchangemedia" -recurse -destination
"C:\\temp\Exchangemedia" -Force

    # rename existing exchange setup log if any
    if (Test-Path C:\\ExchangeSetupLogs\ExchangeSetup.log)
    {
        Rename-Item -Path C:\\ExchangeSetupLogs\ExchangeSetup.log -NewName
ExchangeSetup_old.log -Force
    }
    # Build cmdline syntax according to the scenario to run PrepareAD which inturn would run
PrepareSchema
    $prepareAD = Start-Job -ScriptBlock { Param($Organization) cmd.exe /c
c:\\temp\\Exchangemedia\\setup.com /PrepareAD /OrganizationName:($Organization) } -ArgumentList
$Org_name;
```

PrepareAD.ps1 (Prepare Schema and AD)

```
    Logger $Org_name


    # We need to check the ExchangeSetuplogs directory so we can poll from it
    for ($i=0;$i -lt 10;$i++)
    {
        if ((Test-Path -Path "C:\ExchangeSetupLogs\ExchangeSetup.log") -eq $false)
        {
            Logger "Waiting for PrepareAD to start on this server"
            Sleep 30
        }
        else
        {
            break
        }
    }


    if ((Test-Path -Path "C:\ExchangeSetupLogs\ExchangeSetup.log") -eq $false) { return $false
}

        ## Poll the log file to make sure the command has finished
        for ($j=0;$j -lt 30;$j++)
    {
        if ((Get-Content -Path "C:\ExchangeSetupLogs\ExchangeSetup.log")[((Get-Content -Path
"C:\ExchangeSetupLogs\ExchangeSetup.log").count -3)]  -notlike "*setup operation completed
successfully.")
        {
           Logger "Waiting for PrepareAD to complete"
           sleep 60
        }
        else
        {
            break
        }
    }

    if ((Get-Content -Path "C:\ExchangeSetupLogs\ExchangeSetup.log")[((Get-Content -Path
"C:\ExchangeSetupLogs\ExchangeSetup.log").count -3)]  -notlike "*setup operation completed
successfully.")
    {
        $setup_res = $false
    }
    else
    {
        $setup_res = $true;
    }

    if ($setup_res -eq $true)
    {
        Logger "Microsoft Exchange server schema and AD updated successfully";
    }
    else
    {
        Logger "Schema and AD prep did not complete successfully and timed out. please check
the Exchange setup log";
    }
}
PrepareAD
```

InstallExchroles.ps1 (Exchange Install)

```
# Exchange Install Module
Function global:InstallExchange {
    param
```

InstallExchroles.ps1 (Exchange Install)

```
    (
    [parameter(Mandatory=$true,
         ValueFromPipeline=$true,
         HelpMessage="Please mention the role to be installed. ")]
    [String[]]
    [ValidateNotNull()]
    [ValidateSet("Hub Transport", "Client Access Server", "Mailbox", "Unified Messaging", "Edge
Transport")]
    $Role
    ,
    [parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
    [String]
    $Path
    ,
    [parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]

    [System.Management.Automation.PSCredential]
    $Credential
    )
    Set-Variable -Name credential -Value $Credential -Scope global
    Set-Variable -Name share_path -Value $Path -Scope global
    Set-Variable -Name feature -Value $Role -Scope global
    Import-Module .\logger.ps1
    Logger "--------------------------------------------------------------------------------"
    Logger "Running Exchange Install..."
    $nwcred_username = $credential.getnetworkcredential().UserName
    $nwcred_password = $credential.getnetworkcredential().Password
    net use $share_path $nwcred_password /USER:$nwcred_username
    if (Test-Path $share_path) {
        Logger "$path is validated with $($credential.UserName) and Role to be installed is
$Role"
    } else {
        Logger "cannot connect to $path. Please check the path and credentials!"
        exit
    }

       Logger "Running Exchange Server 2010 Setup Process"

       # To start the install process, Copy the Exchange Server 2010 Setup files to the
designated Exchange Server.
       Logger "Copying the Exchange Server 2010 Setup files to the Designated Server"
       Copy-item -Path "$share_path\ExchangeMedia" -recurse -destination
"c:\temp\ExchangeMedia" -Force

       # rename existing exchange setup log if any
       if (Test-Path C:\ExchangeSetupLogs\ExchangeSetup.log) {
        Rename-Item -Path C:\ExchangeSetupLogs\ExchangeSetup.log -NewName ExchangeSetup_old.log
-Force
       }

       # Build the necessary command syntax to initiate the install process of the planned
Exchange Server role
       $Exch = 'cmd /c "C:\temp\ExchangeMedia\setup.com" /M:install /R:'
Foreach ($item in $feature) {
    if (!($Exch.EndsWith(":"))) { $Exch = $Exch + "," }
    if ($item -ieq "Hub Transport") {
            $Exch = $Exch + "H"
    } elseif ($item -ieq "Client Access Server") {
            $Exch = $Exch + "C"
    } elseif ($item -ieq "Mailbox") {
            $Exch = $Exch + "MB"
    } elseif ($item -ieq "Unified Messaging") {
            $Exch = $Exch + "U"
    } elseif ($item -ieq "Edge Transport") {
            $Exch = $Exch + "E"
```

InstallExchroles.ps1 (Exchange Install)

```
    } else {
            Logger "Not a valid Role to install"
            return
    }
}
        # Now write the bat file...
              $Exch | Out-File -Append c:\temp\roles.bat -Encoding "default"
        Invoke-Expression 'c:\temp\roles.bat'
}
InstallExchange
```

SnapDrive and Storage Provisioning (Map_Luns.ps1)

```
##########################################
       ### create and map luns ###
##########################################

Function global:EXCHluncreate
{
param
(
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
[String]
$Path
,
#storage IP, aggr details, no. of luns for log and db, lun size,
[parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]

[System.Management.Automation.PSCredential]
$Credential_share
,
[parameter(Mandatory=$false,
           HelpMessage="credentials for the domain")]

[System.Management.Automation.PSCredential]
$Credential_domain
,
[parameter(Mandatory=$false,
           HelpMessage="credentials for the storage")]

[System.Management.Automation.PSCredential]
$Credential_storage
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="IP address NetApp Controller")]
[System.Net.IPAddress]
$storage_IP
,
[parameter(Mandatory=$true,
            ValueFromPipeline=$true,
            HelpMessage="ControllerName")]
[String]
$controllername
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="Number of disks to be used to create the aggr")]
[int32]
$aggr_disk_count
,
[parameter(Mandatory=$true,
```

SnapDrive and Storage Provisioning (Map_Luns.ps1)

```
           ValueFromPipeline=$true,
           HelpMessage="number of DB luns to be created")]
[int32]
$DB_lun_count
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="size of each DB lun to be created")]
[string]
$DB_lun_size
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="number of log luns to be created")]
[int32]
$log_lun_count
,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="size of each log lun to be created")]
[string]
$log_lun_size
)

if ($credential_domain -eq $null) {
    $credential_domain = $host.ui.PromptForCredential("Need credentials", "Please enter your
domain\user name and password.", "", "")
}
if ($credential_storage -eq $null) {
   $credential_storage = $host.ui.PromptForCredential("Need credentials", "Please enter your
controller user name and password.", "", "")
}

# setting variables to global
Set-Variable -Name credential_share -Value $Credential_share -Scope global
Set-Variable -Name share_path -Value $Path -Scope global
Set-Variable -Name credential_domain -Value $Credential_domain -Scope global
Set-Variable -Name credential_storage -Value $Credential_storage -Scope global
Set-Variable -Name storage_IP -Value $storage_IP -Scope global
Set-Variable -Name aggr_disk_count -Value $aggr_disk_count -Scope global
Set-Variable -Name DB_lun_count -Value $DB_lun_count -Scope global
Set-Variable -Name DB_lun_size -Value $DB_lun_size -Scope global
Set-Variable -Name log_lun_count -Value $log_lun_count -Scope global
Set-Variable -Name log_lun_size -Value $log_lun_size -Scope global
Set-Variable -Name controllername -Value $controllername -Scope global

Import-Module .\logger.ps1
Logger "--------------------------------------------------------------------------------"
Logger "Running map_luns script.."

# preparing for SDW installation
$sdw_share = "$share_path\Pre-req"
$nwcred_username = $credential_share.GetNetworkCredential().UserName
$nwcred_password = $credential_share.GetNetworkCredential().Password
net use $sdw_share $nwcred_password /USER:$nwcred_username
New-Item -ItemType directory -Path c:\sdtemp -Force
Copy-Item -Path "$sdw_share\SnapDrive6.4.1_x64.exe" -Destination "c:\sdtemp" -Force
net use /delete $sdw_share

# sdw installation
$domain_username = $credential_domain.UserName
$domain_password = $credential_domain.GetNetworkCredential().Password
$storage_username = $credential_storage.GetNetworkCredential().Username
$storage_password = $credential_storage.GetNetworkCredential().Password
$sdw_cmd = 'cmd.exe /c c:\sdtemp\SnapDrive6.4.1_x64.exe /s /v"/qn SILENT_MODE=1 /Li
C:\sdtemp\SDInstall.log INSTALLDIR=\"c:\Program Files\NetApp\SnapDrive\"
SVCUSERNAME='+$domain_username+' SVCUSERPASSWORD='+$domain_password+'
SVCCONFIRMUSERPASSWORD='+$domain_password+' SDW_WEBSRV_TCP_PORT=808 SDW_WEBSRV_HTTP_PORT=4098
```

SnapDrive and Storage Provisioning (Map_Luns.ps1)

```
TRANSPORT_PRT_SELECTION=3 TRANSPORT_PRT_PORT=443
TRANSPORT_PROTOCOL_LOGON_USERNAME='+$storage_username+'
TRANSPORT_PROTOCOL_LOGON_PASSWORD='+$storage_password+' IGNORE_COMPMGMT_RUNNING=1"' | Out-File
-FilePath c:\sdtemp\sdw_install.bat -Encoding "default"
$firewall_cmd = "netsh advfirewall firewall add rule name=`"SnapDrive`" dir=in
program=`"C:\Program Files\NetApp\SnapDrive\SWSvc.exe`" security=authnoencap action=allow" |
Out-File -Append c:\sdtemp\sdw_install.bat -Encoding "Default"

Logger "Installing SnapDrive..."
#install snapdrive
Invoke-Expression 'c:\sdtemp\sdw_install.bat'
Logger "SnapDrive installation completed."

# ensuring iscsi service is started on the host
set-service msiscsi -startup automatic
(Get-WmiObject -class Win32_Service -Filter
"Name='msiscsi'").InvokeMethod("StartService",$null)
Logger "iscsi service running.."

Import-Module DataONTAP

Connect-NaController $storage_IP -Credential $credential_storage
Logger "connected to $storage_IP"

# get host iqn information
$iqn = Get-NaHostIscsiAdapter| Select-Object -ExpandProperty Iqn
if (-not $iqn) {write-warning "no iscsi adapter found";break}

# find the Igroup for this host
$igroup_name = "viaRPC."+$iqn
$Igroup = Get-NaIgroup | Where-Object {($_.Initiators|Select -Expand InitiatorName) -contains
$iqn}
If (-Not $Igroup)
{
    $igroup = New-NaIgroup -Name $igroup_name -Protocol iscsi -Type windows | Add-
NaIgroupInitiator -Initiator $iqn
}

# establish iscsi connection
## ENSURE ISCSI IS LICENSED ON THE STORAGE SYSTEM##
Enable-NaIscsi -ErrorAction silentlycontinue
$PortalList = get-wmiobject -computer $ENV:COMPUTERNAME -namespace root\wmi -query "select
portalinformation from MSiSCSI_PortalInfoClass"
$iSCSIPortID = ($PortalList.PortalInformation | ? {$_.IpAddr.IPV4Address -match
$IPAddress.Address} | select port)
$portnum = $iSCSIPortID.port
$target_iqn = Get-NaIscsiNodeName

#using sdcli to establish iscsi session
$sdcli_cmd =  & 'C:\Program Files\NetApp\SnapDrive\SDCLI.exe' iscsi_initiator establish_session
-m $env:COMPUTERNAME -h 0 -hp $portnum -t $target_iqn -np $storage_IP.IPAddressToString 3260


$aggr = New-NaAggr -Name "EXCH_NVA_test" -DiskCount $aggr_disk_count -ErrorAction stop #
specify mandatory params
for ($i=1;$i -le 12;$i++) {
    $aggr = Get-NaAggr "EXCH_NVA_test"
    if (!($aggr.state -eq "online")) {
        logger "aggr creation in progress..."
        Start-Sleep -Seconds 900
    } else { break }
}
if (!($aggr.state -eq "online")) {
    logger "Please initialise Disks and then run this script"
    exit
}

#create and map DB vol
```

SnapDrive and Storage Provisioning (Map_Luns.ps1)

```
for ($DB=1 ; $DB -le $DB_lun_count; $DB++) {
    $DB_vol = new-navol -Aggregate $aggr -Size $DB_lun_size -Name "Exch_DB_test_$DB" -
SpaceReserve none -ErrorAction stop
    $DB_lun= New-NaLun -Path /vol/$DB_vol/$DB_vol -Type windows_2008 -Size $DB_lun_size -
Unreserved -ErrorAction Stop
    Add-NaLunMap -Path $DB_lun -InitiatorGroup $Igroup.InitiatorGroupName
    Start-NaHostDiskRescan; Wait-NaHostDisk -SettlingTime 5000
    Wait-NaHostDisk -ControllerLunPath $DB_lun.Path -ControllerName $controllername | New-
NaHostVolume
    logger "mapped $DB_lun"
}
# create and map log lun
for ($log=1;$log -le $log_lun_count;$log++) {
    $log_vol = new-navol -Aggregate $aggr -Size $log_lun_size -Name "Exch_log_test_$log" -
SpaceReserve none -ErrorAction silentlycontinue
    $log_lun = New-NaLun -Path /vol/$log_vol/$log_vol -Type windows_2008 -Size $log_lun_size -
Unreserved -ErrorAction Stop
    Add-NaLunMap -Path $log_lun -InitiatorGroup $Igroup.InitiatorGroupName
    Start-NaHostDiskRescan; Wait-NaHostDisk -SettlingTime 5000
    Wait-NaHostDisk -ControllerLunPath $log_lun.Path -ControllerName $controllername| New-
NaHostVolume
    logger "mapped $log_lun"
}
logger "Lun Creation and Mapping Operation Completed."
}
EXCHluncreate
```

SME Install (SMEInstall.ps1)

```
Function SMEInstall
{
param
(
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
[String]
$Path
,
#storage IP, aggr details, no. of luns for log and db, lun size,
[parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]

[System.Management.Automation.PSCredential]
$Credential_share
,
[parameter(Mandatory=$true,
           HelpMessage="credentials for the domain")]

[System.Management.Automation.PSCredential]
$Credential_domain
<#,
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="License")]
[String]
$serial#>
)

Set-Variable -Name credential_share -Value $Credential_share -Scope global
Set-Variable -Name share_path -Value $Path -Scope global
Set-Variable -Name credential_domain -Value $Credential_domain -Scope global
#Set-Variable -Name serial_num -Value $serial -Scope global
Import-Module .\logger.ps1
Logger "--------------------------------------------------------------------------"
```

SME Install (SMEInstall.ps1)

```
Logger "Running SMEInstall script..."
$SMEshare = "$share_path\Pre-Req"
$nwcred_username = $credential_share.GetNetworkCredential().UserName
$nwcred_password = $credential_share.GetNetworkCredential().Password
net use $SMEshare $nwcred_password /USER:$nwcred_username

$SME_install = "$share_path\Pre-Req\SME6.0.2R1_x64.exe /S /v`"AGREETOLICENSE=Yes SILENT_MODE=1
SVCUSERNAME="+$credential_domain.username+"
SVCUSERPASSWORD="+$credential_domain.getnetworkcredential().password+"
SVCCONFIRMUSERPASSWORD="+$credential_domain.getnetworkcredential().password+" /qb`"" | Out-File
c:\temp\SMEinstall.bat -Encoding Default
Logger "executing $SME_install..."
Invoke-Expression 'C:\temp\SMEinstall.bat'
Logger "SME installation completed."
}
SMEInstall
```

SMBR Install (SMBRInstall.ps1)

```
Function SMBRInstall
{
param
(
[parameter(Mandatory=$true,
           ValueFromPipeline=$true,
           HelpMessage="share")]
[String]
$Path
,
#storage IP, aggr details, no. of luns for log and db, lun size,
[parameter(Mandatory=$true,
           HelpMessage="credentials for the share")]

[System.Management.Automation.PSCredential]
$Credential_share

)

Set-Variable -Name credential_share -Value $Credential_share -Scope global
Set-Variable -Name share_path -Value $Path -Scope global
Import-Module .\logger.ps1
Logger "--------------------------------------------------------------------------------"
Logger "Running SMBRinstall script..."

$SMBRshare = "$share_path\Pre-Req"
$nwcred_username = $credential_share.GetNetworkCredential().UserName
$nwcred_password = $credential_share.GetNetworkCredential().Password
net use $SMBRshare $nwcred_password /USER:$nwcred_username

$SMBR_extract = "$smbrshare\SMBR6.0.3.exe /b`"C:\Temp`" /S" | Out-File c:\temp\SMBRinstall.bat
-Encoding Default
$SMBR_install = "msiexec /package `"C:\Temp\Single Mailbox Recovery 6.0.msi`" /quiet /l*v
C:\NVAlogs\SMBRInstall.log" | Out-File c:\temp\SMBRinstall.bat -Encoding Default -Append
Invoke-Expression 'C:\temp\SMBRinstall.bat'
Logger "SMBR installation complete. Please check C:\NVAlogs\SMBRinstall.log"
}
smbrinstall
```

Logger (Logger.ps1)

```
Function global:Logger
{
```

Logger (Logger.ps1)

```
param
(
[parameter(Mandatory=$false,
            ValueFromPipeLine=$true,
            HelpMessage="text to write in the file")]
[string]
$texttolog
,
[parameter(Mandatory=$false,
            ValueFromPipeLine=$true,
            HelpMessage="path of the file to be written into")]
[String]
$logpath
)

Set-Variable -Name texttolog -Value $texttolog -Scope global
Set-Variable -Name logpath -Value $logpath -Scope global
$timestamp = date
if (!(Test-Path c:\NVAlogs)) { New-Item -ItemType directory -Path c:\NVAlogs }
Write-Host "$texttolog"
Out-File -InputObject "$timestamp $texttolog" -FilePath c:\NVAlogs\NVAlog.log -Encoding default
-Append
}
```

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

NetApp provides no representations or warranties regarding the accuracy, reliability, or serviceability of any information or recommendations provided in this publication, or with respect to any results that may be obtained by the use of the information or observance of any recommendations provided herein. The information in this document is distributed AS IS, and the use of this information or the implementation of any recommendations or techniques herein is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. This document and the information contained herein may be used solely in connection with the NetApp products discussed in this document.

Go further, faster®

www.netapp.com