Hewlett Packard Enterprise

Technical white paper

Check if the document is available in the language of your choice.

DEPLOYING DISTRIBUTED FILE SYSTEM REPLICATION ON HPE SAN STORAGE

CONTENTS

EXECUTIVE SUMMARY

Distributed File System (DFS) and Distributed File System Replication (DFSR) are only two of the possible options to solve real-world file serving and replication needs. Other options include BranchCacheV2, and Azure File Sync, as well as several third-party products and the capability of customers to create their own unique solution. The DFS and DFSR options however are well suited to use SAN-based storage since the weakness in these Microsoft technologies regarding long-term retention and survivability match extremely well with the strengths in both HPE Nimble Storage and HPE Primera/HPE 3PAR Storage snapshot and replication technologies.

Target audience: Presales consultants, solution architects, storage operators, data center managers, enterprise architects, and deployment and implementation engineers.

Document purpose: The purpose of this document is to provide guidance on a technology that has unique benefits to meet file data management requirements, and for implementing a solution with DFSR on HPE SAN storage.

What is DFS?

In simple terms a Distributed File System (DFS) is a collection of folders and files that appear under a single namespace such that the consumer of that file system only needs to know the relative path to the data needed—instead of the intricate workings of the collections of servers that are hosting that data.

The most common namespace is the domain that you are a member of, for instance if your domain name is `{MyCompany}` and if the first DFS root you create is named `{MyData}`, use the share `\\MyCompany\MyData` to gain access. Legacy stand-alone (non-clustered) file service implementations commonly implement the namespace as the hostname of the server that is hosting the share, but this model has the effect of locking your file service to the lifecycle of the operating system it is hosted on. The DFS model breaks this legacy thinking by implementing a distinct step of creating a namespace the file service will utilize that has a lifecycle attached to your domain instead of your individual file server.

Under this namespace Root (`\\MyCompany\MyData`) you could then create folders for different sets of data and those folders could span multiple physical servers, such as `{SEATAC-SMB-04}` or `{SJC-SMB-05}`, and multiple complex shares. Additionally, these folders can span multiple sites and combine data. As an example, the share `{\\SEATAC-SMB-04\FinancialData\Seattle}` and `{\\SJC-SMB-05\FinancialData\SanJose}` could both be accessed from `{\\MyCompany\MyData\FinancialData\Seattle}` or `{\\MyCompany\MyData\FinancialData\SanJose}`. In this case, when a user traverses these folders, they are automatically (seamlessly) given the correct data. FIGURE 1 is a visual representation of a namespace.

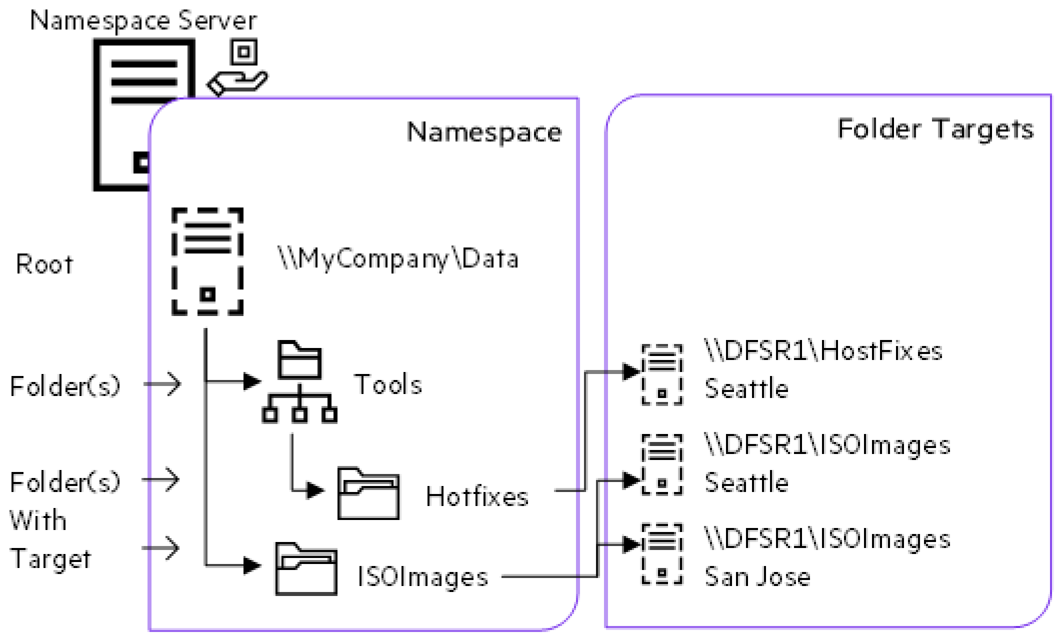


FIGURE 1. Components of a namespace

What is DFSR?

Continuing this example, DFS has not replicated any of the data between the various servers that are servicing the namespace, so users from San Jose accessing the Seattle dataset will traverse the long-distance link to get the data themselves.

Additionally, if each site hosts a set of shares that are common, you can choose to present that data via a common namespace relative location, and the hosts on each site will automatically be directed to the local copy of the `Tools` folder first. This has the advantage of operational simplicity for the users while restricting unneeded site-to-site traffic among the hosts. As an example, if each of the servers contains a share such as `\\SEATAC-SMB-03\LocalTools` then `\\SEATAC-SMB-03\LocalTools` could be replaced with `\\{MyCompany}\{MyData}\LocalTools` that would always connect to the local assets instead of the remote asset. FIGURE 2 is an example of the DFSR process.

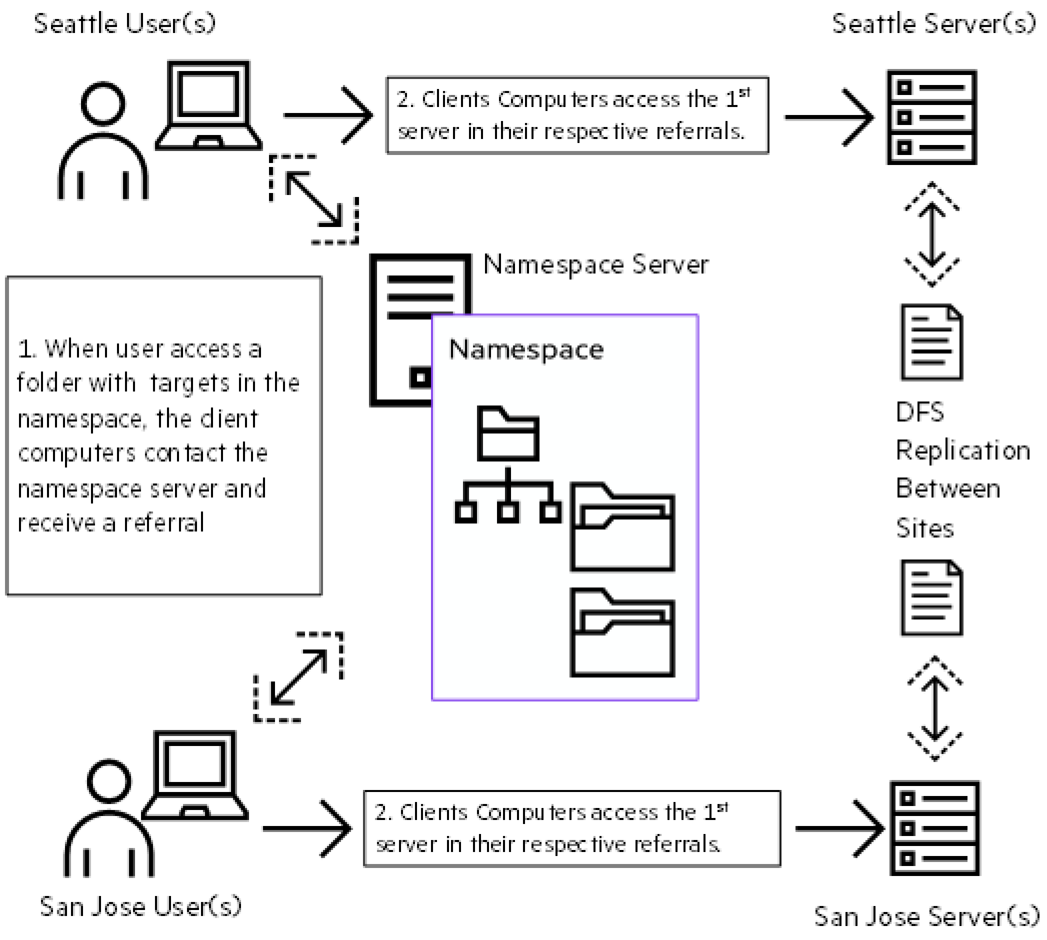

Seattle User(s)
Seattle Server(s)
2. Clients Computers access the 1st server in their respective referrals.
Namespace Server
Namespace
1. When user access a folder with targets in the namespace, the client computers contact the namespace server and receive a referral
DFS Replication Between Sites
2. Clients Computers access the 1st server in their respective referrals.
San Jose User(s)
San Jose Server(s)

FIGURE 2. DFSR process

Without replication being enabled on the above example, each of the shares that are locally accessible will have unique data, and that data is not being synchronized between sites.

This localization is done using the Domain sites feature and is detected and used by DFS and DFSR natively. Be sure that your sites are already defined in Active Directory, because DFS has the option to allow hosts to connect to the lowest cost (defined by site) option as well as the option to exclude targets outside of the client's site. See your AD sites and services management console to make sure that your sites have been defined properly. FIGURE 3 is an example of the Active Directory management console.

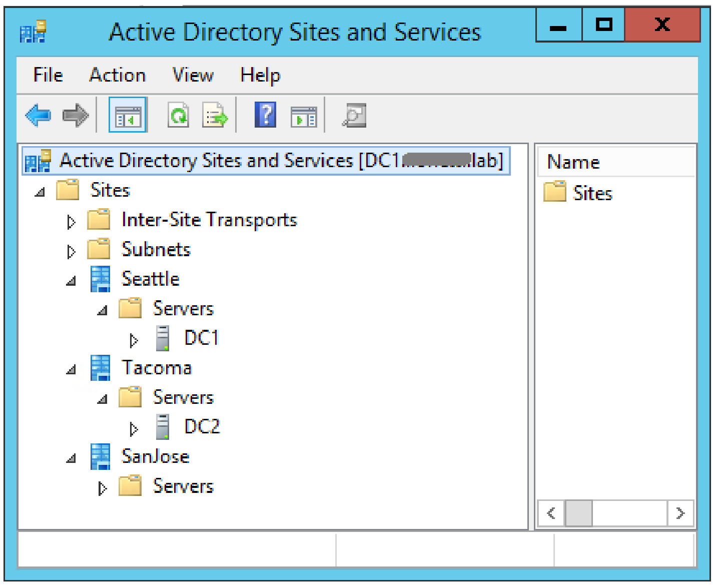


FIGURE 3. Active Directory management console

To offer added protection we add to this design the R (Replication) aspect of DFSR. This is done by replicating a namespace or individual namespace folders from site to site. Each replica copy can be accessed and used in accordance with pre-defined site awareness to make sure that clients connect to the closest (or specific) copy of a specific file. Additionally, DFSR can be configured to allow for one-way replication if you want file changes to only flow one direction. Alternately, you may choose to set the share permissions on the secondary site to prevent write operations to accomplish a similar outcome. The example in FIGURE 4 is the same as the previous FIGURE 2 example, except that the two namespace target folders are going to be constantly kept in sync via DFSR.

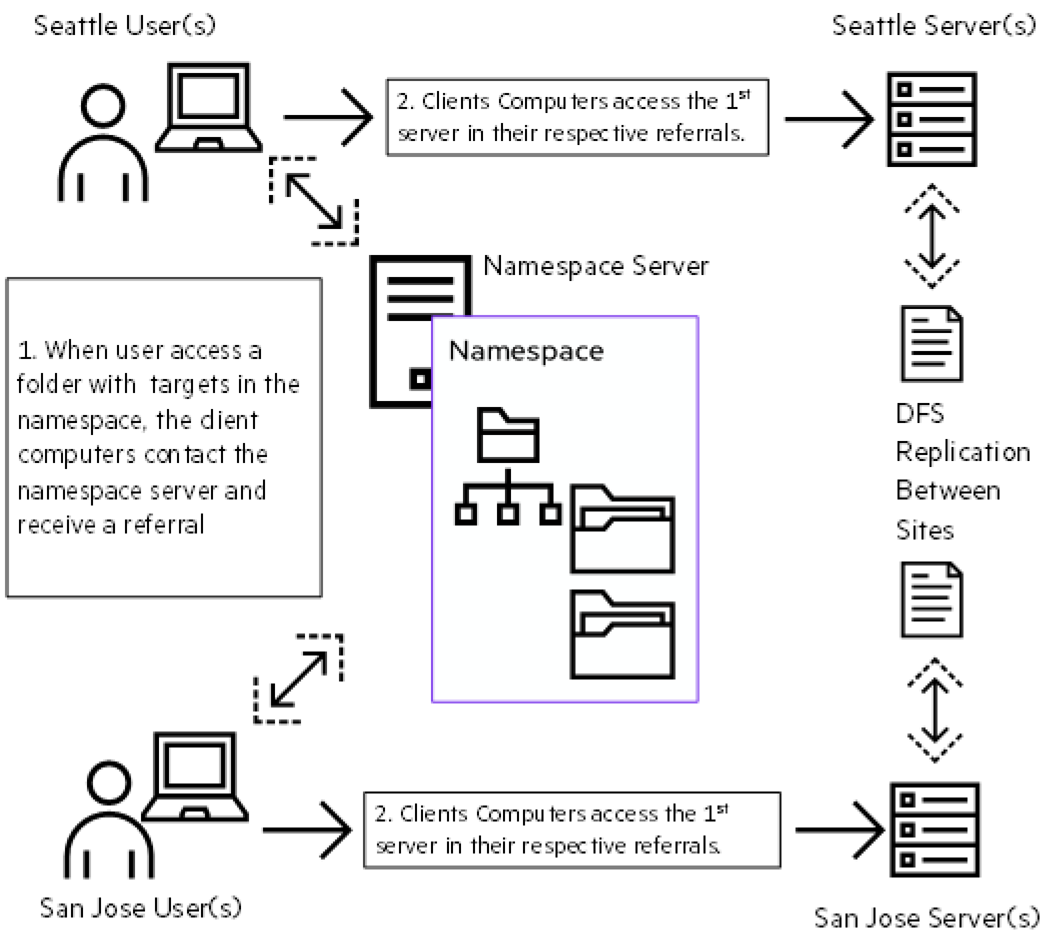

Seattle User(s)
Seattle Server(s)
2. Clients Computers access the 1st server in their respective referrals.
Namespace Server
1. When user access a folder with targets in the namespace, the client computers contact the namespace server and receive a referral
Namespace
DFS Replication Between Sites
2. Clients Computers access the 1st server in their respective referrals.
San Jose User(s)
San Jose Server(s)

FIGURE 4. DFS Replication process

The strengths of DFSR over other forms of replication

The main strength of DFSR is that all copies of the data are under the customer's custodial control; that is, no data-at-rest exists outside of the physical control of the company that deploys the solution. This can be a requirement for extremely security-conscious customers and may not be possible in some cloud-based replication strategies.

Another advantage of DFSR is that it allows for the secondary sites to be pre-seeded with the dataset to be synchronized via array-based replications technologies such as those available in HPE Primera, HPE Nimble, and HPE 3PAR storage technologies.

Additionally, the default Server Message Block (SMB) clients for Windows as well as Linux® clients using SAMBA V4+ natively adhere to DFSR links and referrals as does Apple Mac using OS X Lion or newer.

DFSR also works with File Services Resource Manager (FSRM), which allows for complex rules on your datasets, such as auditing, automatic classification (by age, or content, or group), tier, and so on. There are a few considerations to follow when using FSRM and they are discussed later in this paper.

DFSR is also Windows Failover Cluster (WFC) aware/integrated. This allows any site to host the DFS namespace on a highly available infrastructure and survive single-point-of-failure scenarios. Each site participating in DFSR may be hosted on a WFC, or hosted on a standalone server, and the entire cross-site DFSR implementation can be a combination of WFCs as well as standalone servers.

Each site under DFSR does not need to mirror the entire dataset and can be configured uniquely for each site in a multisite environment based on the needs of that particular environment.

What DFSR is not

DFSR does not have a file-locking function. If a file is modified from two locations, each client assumes they have saved their changes, but the last saved version of the file will win. To offer some version of file locking, you would need to deploy a different type of architecture such as BranchCacheV2 or Azure File Sync, which both offer this functionality.

Additionally, DFSR is not a pre-built appliance that self-configures and guesses your optimal setup. The product is very powerful and has many very powerful options. You will have to make a number of decisions based on your business needs and expectations of how you want DFSR to operate. As an example, there are several valid source and destination models, and they all have different situations where they are more or less suited. DFSR supports Full Mesh Model, Spoke and Hub Model, Point to Point Model, and Ring Model topologies.

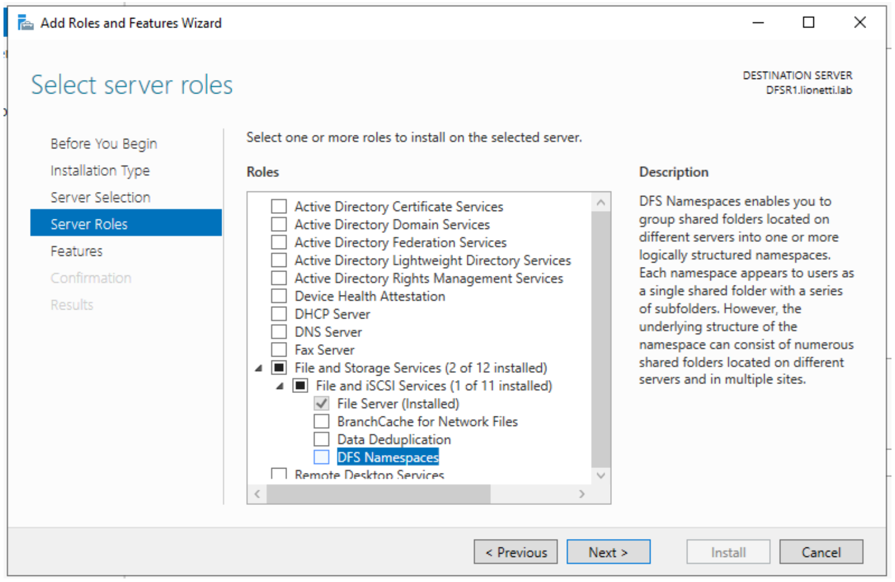
Other similar solutions

Azure File Sync (AFS) is a somewhat similar solution space, but has a hard dependency on the Azure Public Cloud, and requires a monthly fee based on the storage capacity being protected. In AFS, you download and execute an agent on your Windows file server against a specific share. The agent will slowly copy the dataset to the cloud. To add more servers to the AFS from another site (Site 2) you add the AFS agent to those file servers and host the same share. When a user on Site 2 requests a file, if that file does not exist, the file server requests the file from the cloud. If that file has not yet migrated to the cloud from Site 1, the request is passed to the Site 1 server. This file retrieval act will populate the cloud with that file for future use, as well as on Site 2. Because a communication channel exists between Site 2 and the cloud, and the cloud and Site 1, file locks can be set and honored. Note that with no other actions, the share located on Site 1 will completely copy itself to the cloud as a background task. Additionally, Site 2 can be either a full copy of the cloud dataset, or just a cached subset of the full cloud dataset.

BranchCacheV2 is a similar solution but differs from DFSR in that it only allows for a Hub and Spoke model. However, the remote sites use the primary site as the file-lock manager to prevent file conflicts, but requires that the primary site be operational. This method of replication is useful to lower the latency of a dataset to many remote sites, but it is expected that the remote sites do not contain a full backup dataset to the original, so this is commonly not thought of or used as a backup technology. This solution does not require any cloud-based recurring subscription and all copies of the customer data reside only on the customer site.

STEPS TO INSTALL DFS

This paper assumes a number of factors that are most common, such as the fact that the customer is using a modern Active Directory (AD) and has Domain Name Services (DNS) integrated with the AD environment. FIGURE 5 is an example of the **Server Roles** in the **Add Roles and Features Wizard**.



Add Roles and Features Wizard

Select server roles

DESTINATION SERVER
DFSR1.lionetti.lab

Before You Begin
Installation Type
Server Selection
Server Roles
Features
Confirmation
Results

Select one or more roles to install on the selected server.

Roles

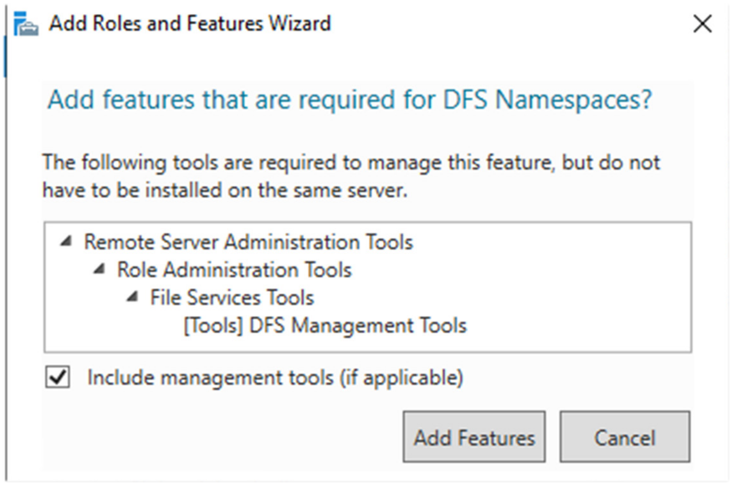
- ☐ Active Directory Certificate Services
- ☐ Active Directory Domain Services
- ☐ Active Directory Federation Services
- ☐ Active Directory Lightweight Directory Services
- ☐ Active Directory Rights Management Services
- ☐ Device Health Attestation
- ☐ DHCP Server
- ☐ DNS Server
- ☐ Fax Server
- ■ File and Storage Services (2 of 12 installed)
  - ■ File and iSCSI Services (1 of 11 installed)
    - ☑ File Server (Installed)
    - ☐ BranchCache for Network Files
    - ☐ Data Deduplication
    - ☐ DFS Namespaces
- ☐ Remote Desktop Services

Description

DFS Namespaces enables you to group shared folders located on different servers into one or more logically structured namespaces. Each namespace appears to users as a single shared folder with a series of subfolders. However, the underlying structure of the namespace can consist of numerous shared folders located on different servers and in multiple sites.

FIGURE 5. "Select server roles" in the "Add Roles and Features Wizard"

After the **Server Roles** are specified, a pop-up window appears for you to validate what will be installed, as shown in FIGURE 6.


Add Roles and Features Wizard

Add features that are required for DFS Namespaces?

The following tools are required to manage this feature, but do not have to be installed on the same server.

- Remote Server Administration Tools
  - Role Administration Tools
    - File Services Tools
      - [Tools] DFS Management Tools

☑ Include management tools (if applicable)

FIGURE 6. Installation validation of required features

After the installation is complete, you will find the DFS Management Tool on the Server manager list, as shown in FIGURE 7.

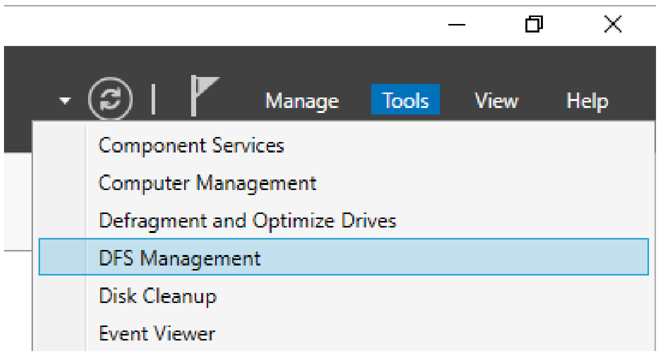


FIGURE 7. "Tools" tab with "DFS Management" highlighted

Perform the following steps to configure the namespace server in the DFS Management tool.

1. Create or add the first namespace, as demonstrated in FIGURE 8.

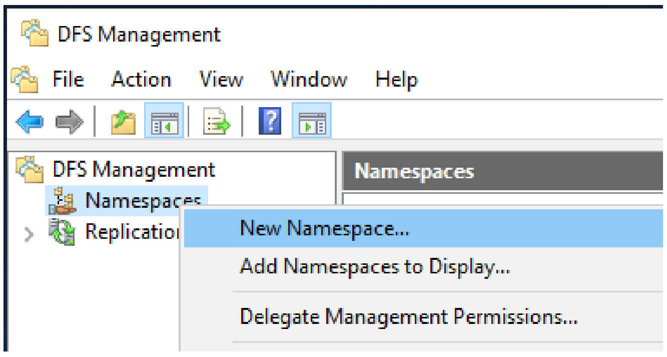
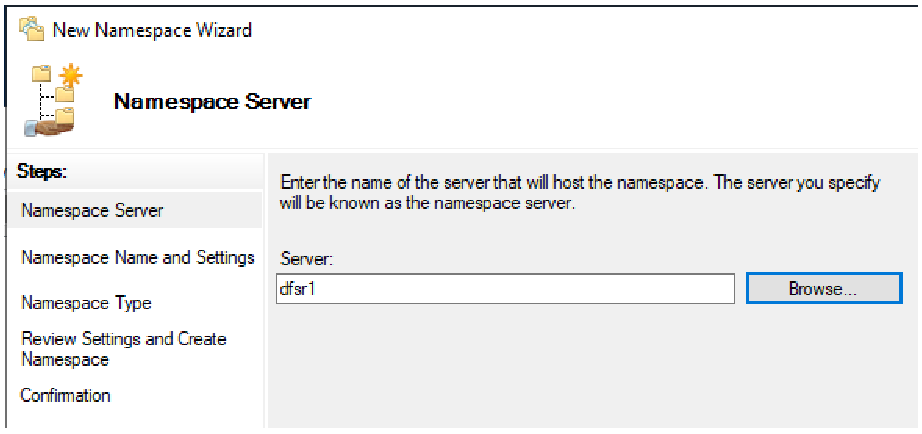


FIGURE 8. "New Namespace" highlighted in the "DFS Management" GUI

2. Enter the name of the server that will host the namespace, illustrated in FIGURE 9.

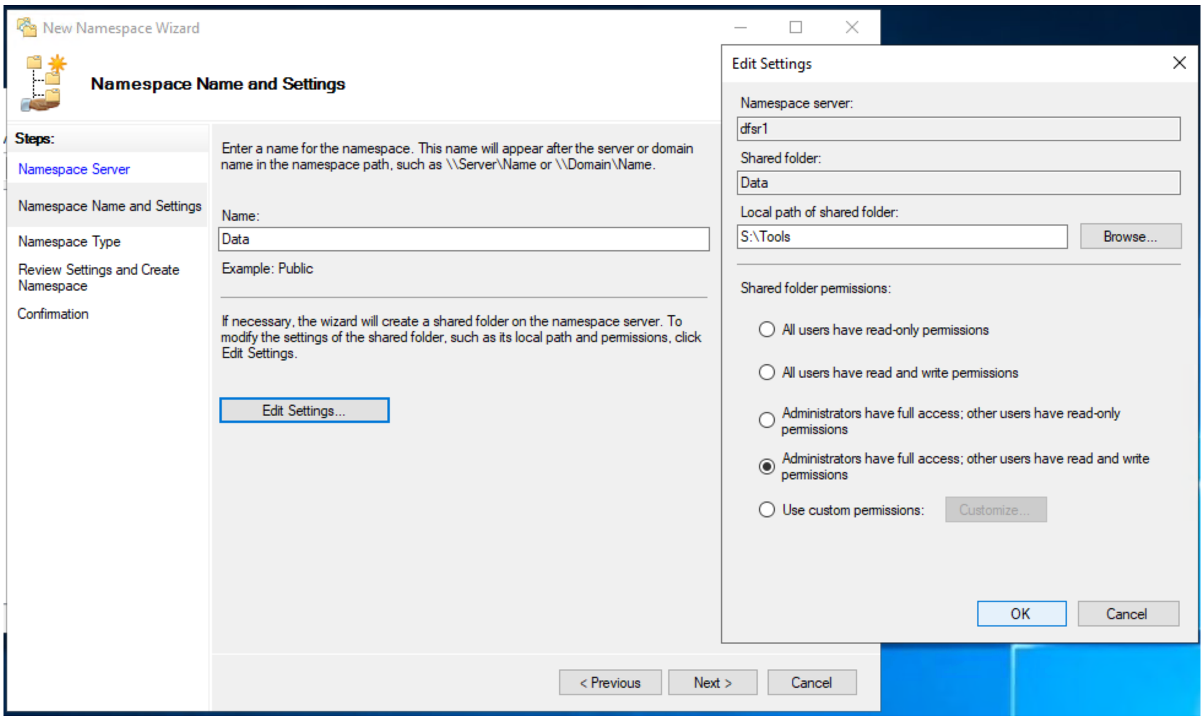


Enter the name of the server that will host the namespace. The server you specify will be known as the namespace server.

Server: dfsr1

FIGURE 9. "Server" field in "Namespace Server" window to enter the server name

3. In the **Namespace Name and Settings** window, click the **Edit Settings...** button and enter the namespace server criteria (FIGURE 10).



Enter a name for the namespace. This name will appear after the server or domain name in the namespace path, such as \\Server\Name or \\Domain\Name.

Name: Data

Example: Public

If necessary, the wizard will create a shared folder on the namespace server. To modify the settings of the shared folder, such as its local path and permissions, click Edit Settings.

Namespace server: dfsr1

Shared folder: Data

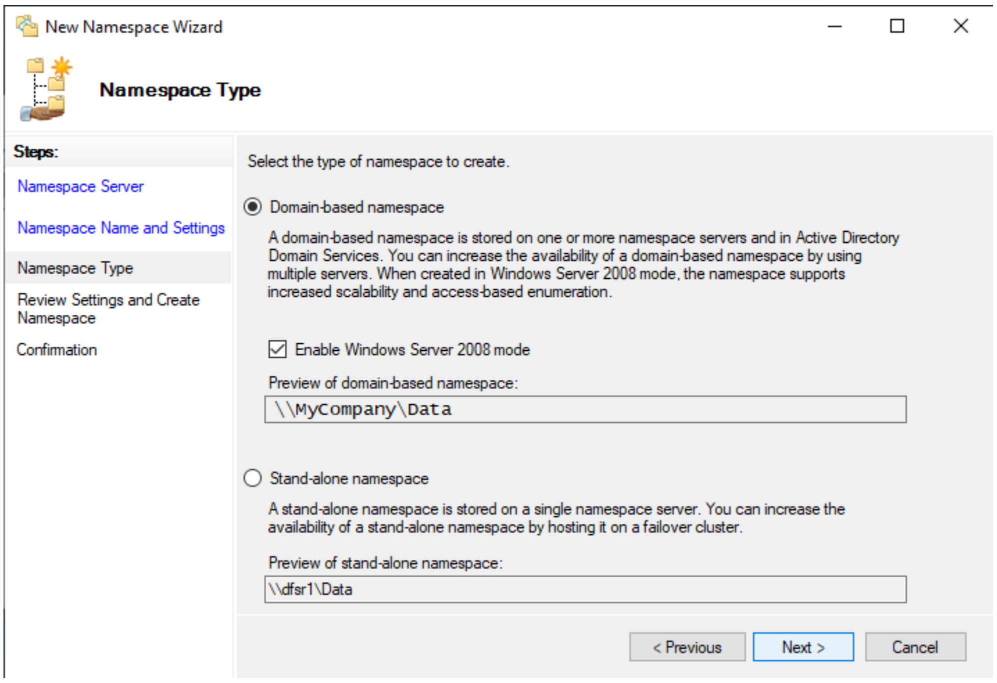
Local path of shared folder: S:\Tools

Shared folder permissions:

- All users have read-only permissions
- All users have read and write permissions
- Administrators have full access; other users have read-only permissions
- [✓ Administrators have full access; other users have read and write permissions]
- Use custom permissions:

FIGURE 10. Editing namespace server settings

4. Select **Domain-based namespace**, as illustrated in FIGURE 11.



Select the type of namespace to create.

● Domain-based namespace

A domain-based namespace is stored on one or more namespace servers and in Active Directory Domain Services. You can increase the availability of a domain-based namespace by using multiple servers. When created in Windows Server 2008 mode, the namespace supports increased scalability and access-based enumeration.

☑ Enable Windows Server 2008 mode

Preview of domain-based namespace:

```
\\MyCompany\Data
```

○ Stand-alone namespace

A stand-alone namespace is stored on a single namespace server. You can increase the availability of a stand-alone namespace by hosting it on a failover cluster.

Preview of stand-alone namespace:

```
\\dfsr1\Data
```

FIGURE 11. Selecting the "Domain-based namespace" type

After the namespace server settings have been created, the replication request needs to make its way from each DFSR server. This can be accelerated by using the following PowerShell command from each node;

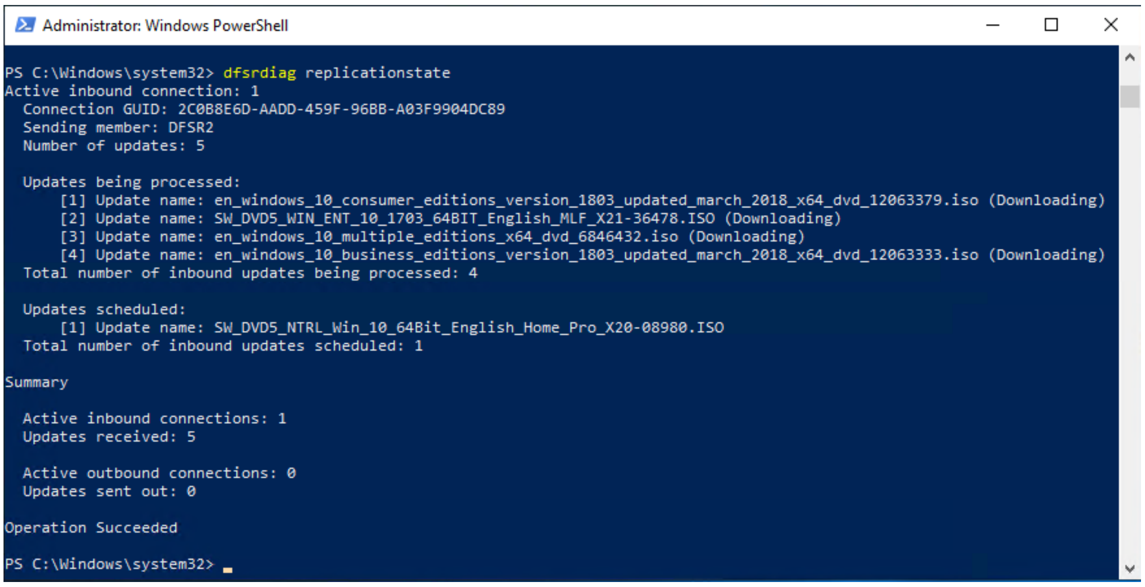
You can force replication immediately by using DFS Management, You can also force replication by using the PowerShell cmdlet `Sync-DfsReplicationGroup` or the `Dfsrdiag SyncNow` command. You can force polling by using the `Update-DfsrConfigurationFromAD`' cmdlet or the `Dfsrdiag PollAD` command.

```
PS:> Dfsrdiag PollAD /member:dfsr1

PS:> Dfsrdiag PollAD /member:dfsr2
```

After the replication has started, which could take 15+ minutes to start, you can monitor it using the following command:

```
PS:> Dfsrdiag ReplicationState
```

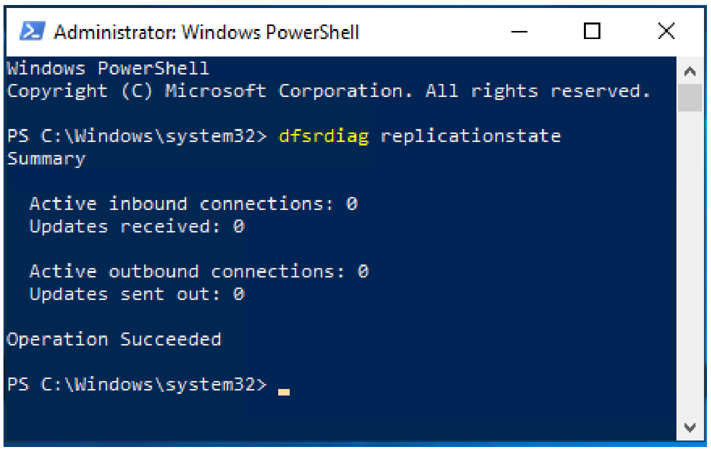
FIGURE 12 is an example of the output you should receive during the initial replication;



```
PS C:\Windows\system32> dfsrdiag replicationstate
Active inbound connection: 1
  Connection GUID: 2C0B8E6D-AADD-459F-96BB-A03F9904DC89
  Sending member: DFSR2
  Number of updates: 5

  Updates being processed:
      [1] Update name: en_windows_10_consumer_editions_version_1803_updated_march_2018_x64_dvd_12063379.iso (Downloading)
      [2] Update name: SW_DVD5_WIN_ENT_10_1703_64BIT_English_MLF_X21-36478.ISO (Downloading)
      [3] Update name: en_windows_10_multiple_editions_x64_dvd_6846432.iso (Downloading)
      [4] Update name: en_windows_10_business_editions_version_1803_updated_march_2018_x64_dvd_12063333.iso (Downloading)
  Total number of inbound updates being processed: 4

  Updates scheduled:
      [1] Update name: SW_DVD5_NTRL_Win_10_64Bit_English_Home_Pro_X20-08980.ISO
  Total number of inbound updates scheduled: 1

Summary

  Active inbound connections: 1
  Updates received: 5

  Active outbound connections: 0
  Updates sent out: 0

Operation Succeeded

PS C:\Windows\system32> _
```

FIGURE 12. PowerShell example of initial replication

If no current replication work is currently occurring, the following will be displayed (FIGURE 13).



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> dfsrdiag replicationstate
Summary

  Active inbound connections: 0
  Updates received: 0

  Active outbound connections: 0
  Updates sent out: 0

Operation Succeeded

PS C:\Windows\system32> _
```

FIGURE 13. PowerShell example of no replication occurring

Because the topology is defined by the customer, for redundancy, the DFSR nodes can exist in the same site and mirror each other's datasets, and the DFSR nodes can exist on remote sites as well. There is no limitation on the types of storage that can be used to support DFSR; however, some of the following advanced topics might require Intelligent Shared Storage.

Advanced DFSR – Using Windows Failover Clustering

The most common use of Intelligent Shared Storage when deploying DFSR is to support highly available SMB shares hosted on Windows Failover Clusters. When deploying an SMB share on a cluster, you have a few distinct advantages: The file sharing service being clustered does not go down due to service packs (using Cluster Aware Updating) and single servers do not fail.

Because the cluster owns the domain name instead of the node, a single multi-node cluster can host or consolidate many file sharing services. For example, the share named `\\MyFileService1\MyShare1` can be hosted on the same cluster as `\\MyFileService2\MyShare2` and each of these `{MyFileService1, MyFileService2}` can be unique DNS records and unique IP addresses. It is very common for customers to consolidate file services from throughout a company onto a single failover cluster using the free Windows Storage Migrator service, which can accomplish this without having to update clients' expected mount points or security settings. This consolidation allows a company's IT department to centralize governance and protection of the data that is crucial.

Additionally, the consolidation allows obsolete hardware across an organization to re-home its hosted data to modern hardware. This modern Windows Failover Cluster (WFC) allows for additions of new nodes and the eviction of old nodes without affecting service uptime so the act of refreshing hardware over time changes from a difficult, costly experience to regular business activity that can be done without notice.

With the addition of DFSR, we now have the ability to deploy redundancy in the file service by mirroring our data between a set of servers on the same site. This differs in two primary ways. First, DFSR provides a data redundancy feature that requires the deployment of twice the amount of raw storage, because each DFSR server needs its own dataset. This extra storage also means that the DFSR servers can consume a significant amount of storage bandwidth keeping these copies in sync.

This process is not needed for WFC shares, because if the server fails, another server in the cluster continues operating the service from the same set of data using the WFC failover process. This means that the initial sync process, the background network consumption, and the non-synchronous nature of the copy are all completely eliminated.

However, DFSR is a far superior solution for mirroring the data between sites since WFCs have inherent limitations related to distance/latency. The DFSR code is cluster aware but does not require that any DFSR instance be either standalone or clustered, and you could host the data on Site 1 using a WFC, which mirrors to a distant Site 2 that is hosted on a standalone file server. FIGURE 14 is an example where the primary site is hosted on shared storage on a Windows Failover Cluster, while the branch office only needs a single standalone server.

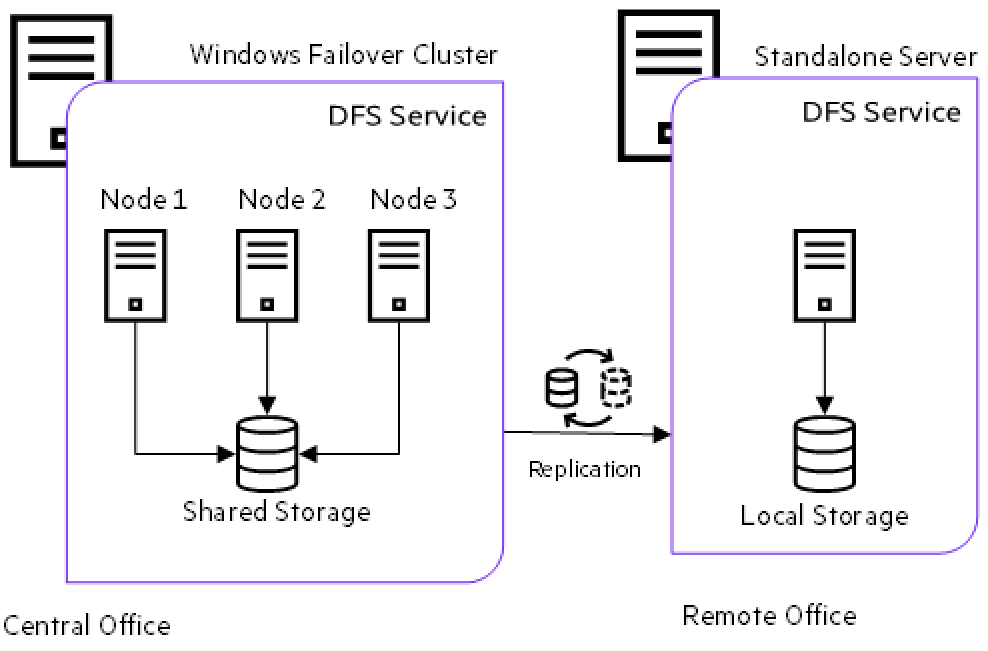


FIGURE 14. Example of primary site hosted on shared storage

Advanced DFSR – Deduplication and pre-seeding copies

This disk consumption used by standalone DFSR servers can be mitigated by hosting the file share volumes that are presented to the DFSR servers from the same HPE SAN storage that enables deduplication, because these copies can easily be deduplicated against each other.

Advanced DFSR – Pre-seeding copies

Additionally, the act of pre-seeding a secondary copy for an additional DFSR server can be done using a snapshot clone, so the initial sync process can be eliminated.

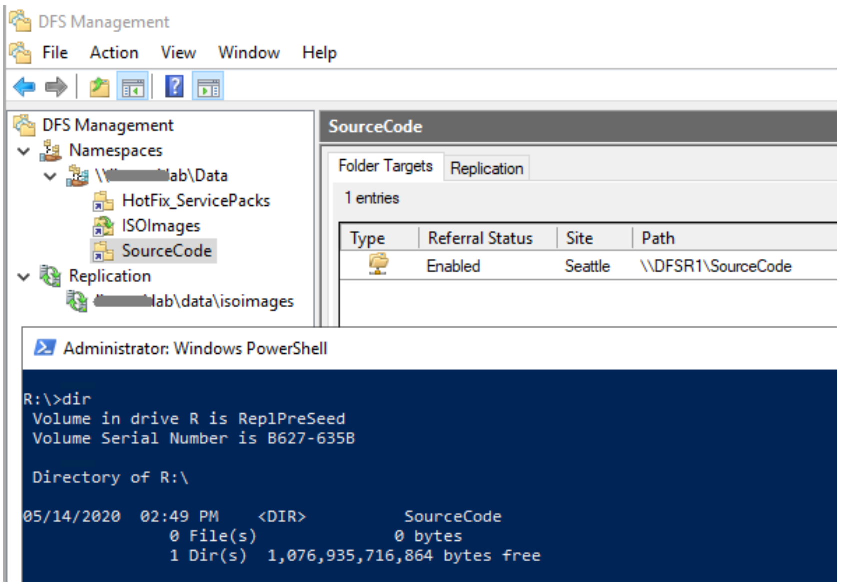
This pre-seeding process is not needed for WFC-deployed DFSR, because only a single copy need exist. However, array-to-array replication can be used to populate a secondary site where a WFC cannot span. The initial sync process using array-based replication is significantly faster than DFSR can natively replicate the dataset. A replication process from the primary site to the secondary site can be asynchronous since the first step after pre-seeding the second site is for each server to checksum the files to determine what files differ and to only send that small subset. This replication can be easily accomplished using either HPE 3PAR Remote Copy or HPE Nimble Storage replication.

In the following step-by-step instructions, we will create a share on DFSR1 named “SourceCode,” which we intend to replicate to DFSR2 using array-based replication for quick seeding the secondary site.

The overall steps are to create a volume that houses ONLY the `SourceCode` folder that is being shared from `S:\SourceCode`. And, at the start of this process, the downstream server does not have a `\SourceCode` folder, drive, or share.

1. Create a replication group and a replicated folder, then add a server as a member of that topology (but no partners, yet). This will be the “upstream” (source) server.

   The FIGURE 15 example shows how the `R:\SourceCode` share has been exposed in DSFR under the default namespace user in the `SourceCode` folder.



Type	Referral Status	Site	Path
	Enabled	Seattle	\\DFSR1\SourceCode

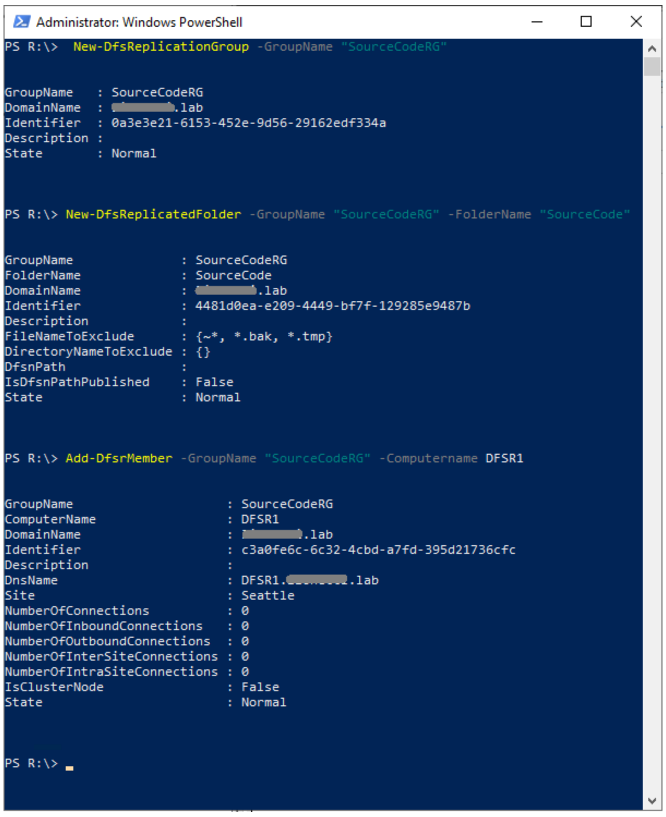
```
R:\>dir
 Volume in drive R is ReplPreSeed
 Volume Serial Number is B627-635B

 Directory of R:\

05/14/2020  02:49 PM    <DIR>          SourceCode
               0 File(s)              0 bytes
               1 Dir(s)  1,076,935,716,864 bytes free
```

FIGURE 15. DFSR-exposed "R:\SourceCode” share

A replication group was created without connecting to the downstream DFSR server. This replication group will only contain the upstream source. An upstream folder was added to the replication group to replicate, as well as the server that hosts this folder.

```
PS:> New-DfsReplicationGroup -GroupName “SourceCodeRG”
PS:> New-DfsReplicatedFolder -GroupName “SourceCodeRG” -FolderName “SourceCode”
PS:> Add-DfsrMember -GroupName “SourceCodeRG” -ComputerName DFSR1
PS:> Set-DfsrMembership -GroupName “SourceCodeERG” -FolderName “SourceCode” `
     -ContentPath “R:\SourceCode” -ComputerName DFSR1 -PrimaryMember $True
PS:> UpdateDfsrConfigurationFromAD -ComputerName DFSR1 -verbose
```

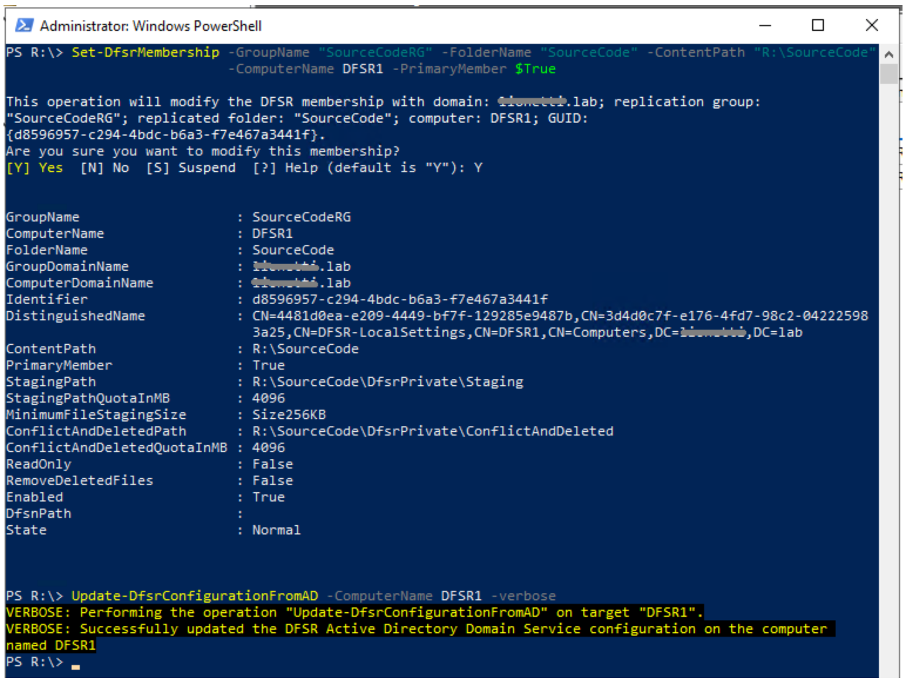
FIGURE 16 illustrates the creation of a new replication folder using PowerShell.


Administrator: Windows PowerShell

```
PS R:\>  New-DfsReplicationGroup -GroupName "SourceCodeRG"


GroupName   : SourceCodeRG
DomainName  : [redacted].lab
Identifier  : 0a3e3e21-6153-452e-9d56-29162edf334a
Description :
State       : Normal



PS R:\> New-DfsReplicatedFolder -GroupName "SourceCodeRG" -FolderName "SourceCode"


GroupName              : SourceCodeRG
FolderName             : SourceCode
DomainName             : [redacted].lab
Identifier             : 4481d0ea-e209-4449-bf7f-129285e9487b
Description            :
FileNameToExclude      : {~*, *.bak, *.tmp}
DirectoryNameToExclude : {}
DfsnPath               :
IsDfsnPathPublished    : False
State                  : Normal



PS R:\> Add-DfsrMember -GroupName "SourceCodeRG" -Computername DFSR1


GroupName                    : SourceCodeRG
ComputerName                 : DFSR1
DomainName                   : [redacted].lab
Identifier                   : c3a0fe6c-6c32-4cbd-a7fd-395d21736cfc
Description                  :
DnsName                      : DFSR1.[redacted].lab
Site                         : Seattle
NumberOfConnections          : 0
NumberOfInboundConnections   : 0
NumberOfOutboundConnections  : 0
NumberOfInterSiteConnections : 0
NumberOfIntraSiteConnections : 0
IsClusterNode                : False
State                        : Normal



PS R:\> _
```

FIGURE 16. Creating the new replication folder using PowerShell commands

FIGURE 17 shows setting the replication membership values using PowerShell.

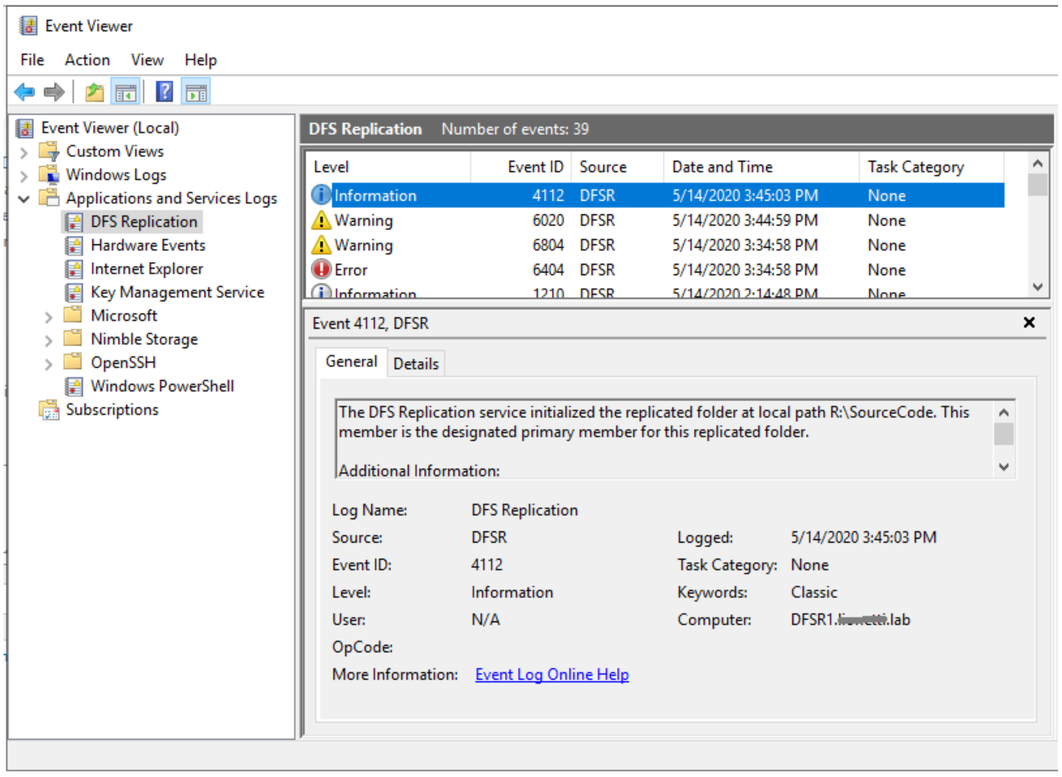

Administrator: Windows PowerShell

```
PS R:\> Set-DfsrMembership -GroupName "SourceCodeRG" -FolderName "SourceCode" -ContentPath "R:\SourceCode"
                           -ComputerName DFSR1 -PrimaryMember $True

This operation will modify the DFSR membership with domain: [redacted].lab; replication group:
"SourceCodeRG"; replicated folder: "SourceCode"; computer: DFSR1; GUID:
{d8596957-c294-4bdc-b6a3-f7e467a3441f}.
Are you sure you want to modify this membership?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y


GroupName                    : SourceCodeRG
ComputerName                 : DFSR1
FolderName                   : SourceCode
GroupDomainName              : [redacted].lab
ComputerDomainName           : [redacted].lab
Identifier                   : d8596957-c294-4bdc-b6a3-f7e467a3441f
DistinguishedName            : CN=4481d0ea-e209-4449-bf7f-129285e9487b,CN=3d4d0c7f-e176-4fd7-98c2-04222598
                               3a25,CN=DFSR-LocalSettings,CN=DFSR1,CN=Computers,DC=[redacted],DC=lab
ContentPath                  : R:\SourceCode
PrimaryMember                : True
StagingPath                  : R:\SourceCode\DfsrPrivate\Staging
StagingPathQuotaInMB         : 4096
MinimumFileStagingSize       : Size256KB
ConflictAndDeletedPath       : R:\SourceCode\DfsrPrivate\ConflictAndDeleted
ConflictAndDeletedQuotaInMB  : 4096
ReadOnly                     : False
RemoveDeletedFiles           : False
Enabled                      : True
DfsnPath                     :
State                        : Normal



PS R:\> Update-DfsrConfigurationFromAD -ComputerName DFSR1 -verbose
VERBOSE: Performing the operation "Update-DfsrConfigurationFromAD" on target "DFSR1".
VERBOSE: Successfully updated the DFSR Active Directory Domain Service configuration on the computer
named DFSR1
PS R:\> _
```

FIGURE 17. Updating the replication membership using PowerShell

2. At this point the DFSR engine is creating a database of the contents of the DFS folder, which will be used to fingerprint each file to know which files have and have not been replicated. Wait for this process to complete and watch the Event Log for an event from "DFS Replication Event" with an ID of 4112.

   FIGURE 18 is an example of determining the replication status using Event Viewer.



DFS Replication Number of events: 39

Level	Event ID	Source	Date and Time	Task Category
Information	4112	DFSR	5/14/2020 3:45:03 PM	None
Warning	6020	DFSR	5/14/2020 3:44:59 PM	None
Warning	6804	DFSR	5/14/2020 3:34:58 PM	None
Error	6404	DFSR	5/14/2020 3:34:58 PM	None
Information	1210	DFSR	5/14/2020 2:14:48 PM	None

Event 4112, DFSR

The DFS Replication service initialized the replicated folder at local path R:\SourceCode. This member is the designated primary member for this replicated folder.

Additional Information:

Log Name:	DFS Replication		
Source:	DFSR	Logged:	5/14/2020 3:45:03 PM
Event ID:	4112	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	DFSR1.[redacted].lab
OpCode:			
More Information:	Event Log Online Help		

FIGURE 18. Event Viewer view of replication events

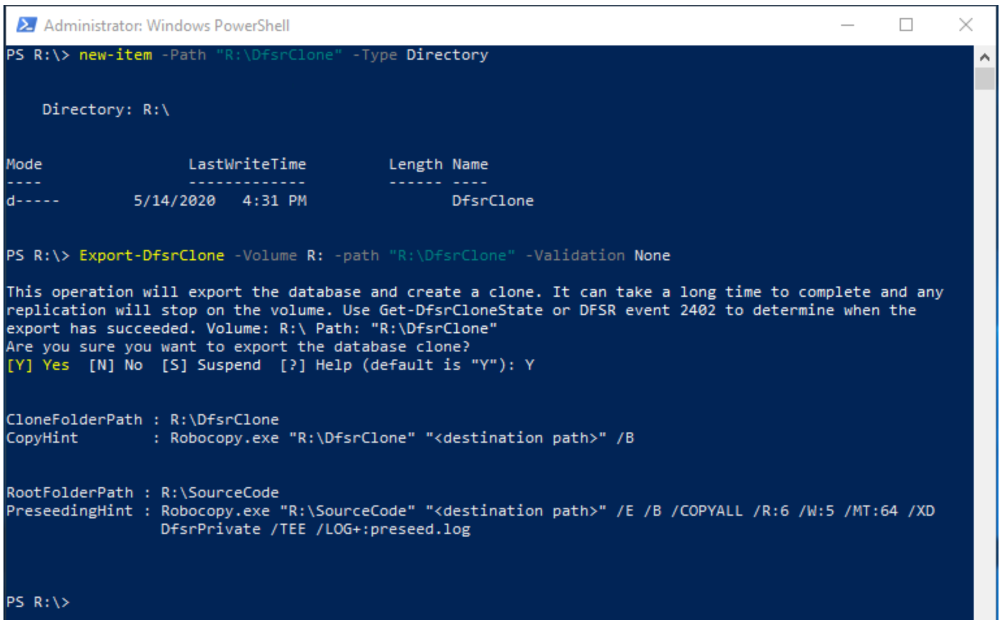
3. After the DB build is complete, the DFSR database can be cloned from the source. You also need to determine which level of validation is required during the database cloning operation. There are three options, chose one of the following:

   – None—This method is by far the fastest, but best suited if the upstream dataset is not changing heavily during the cutover window, as those files changed inside the cutover window might not be updated until they are again modified after the downstream is in sync. The upstream copy does not need to be taken offline, but it is recommended to temporarily make this copy read-only until the cutover is complete.

   – Basic—This method will maintain a hash of the metadata of each file, including the last written time. This method requires that the copy from site to site is perfect since the file contents are not hashed, but will automatically pick up any files that are changed during the cutover period. This method is the default and will work with no other intervention.

   – Full—This method will create a hash for the metadata and contents of each file. This mode is most forgiving in the case of an incomplete or flawed copy but is very slow and not recommended for any dataset over 10 TB in size. Because an array-based perfect copy is being used, this method is not recommended or needed.

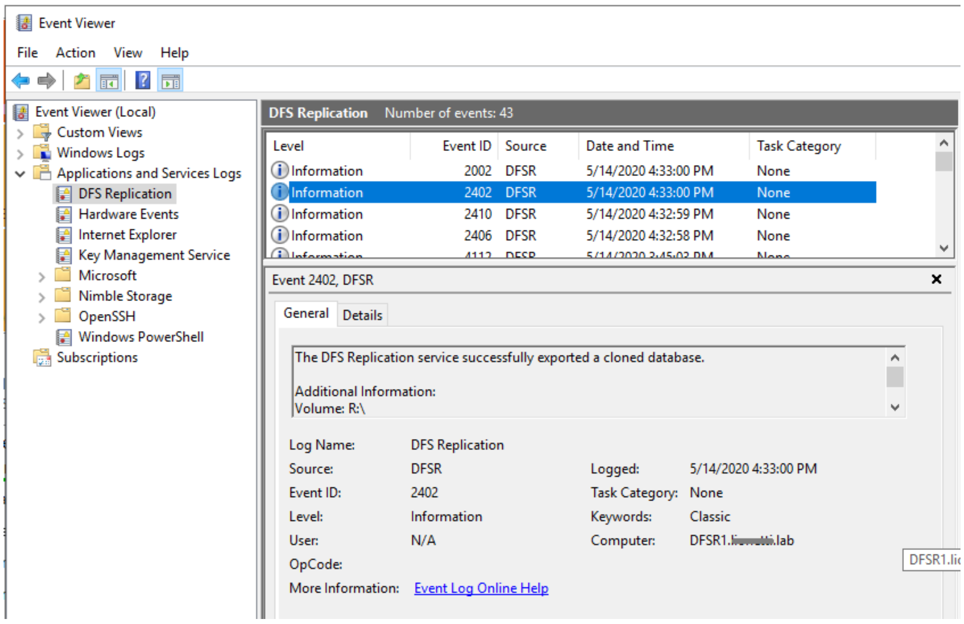
4. Next, create a folder in the volume to contain the cloned DB, which can be imported into the downstream server. FIGURE 19 illustrates adding the folder using PowerShell.



```
PS R:\> new-item -Path "R:\DfsrClone" -Type Directory


    Directory: R:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/14/2020   4:31 PM                DfsrClone


PS R:\> Export-DfsrClone -Volume R: -path "R:\DfsrClone" -Validation None

This operation will export the database and create a clone. It can take a long time to complete and any
replication will stop on the volume. Use Get-DfsrCloneState or DFSR event 2402 to determine when the
export has succeeded. Volume: R:\ Path: "R:\DfsrClone"
Are you sure you want to export the database clone?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y


CloneFolderPath : R:\DfsrClone
CopyHint        : Robocopy.exe "R:\DfsrClone" "<destination path>" /B


RootFolderPath : R:\SourceCode
PreseedingHint : Robocopy.exe "R:\SourceCode" "<destination path>" /E /B /COPYALL /R:6 /W:5 /MT:64 /XD
                 DfsrPrivate /TEE /LOG+:preseed.log



PS R:\>
```

FIGURE 19. Example of the "new-item" command to create the folder to store the cloned DB

5. After the `new-item` command completes, wait until a DFS Replication Event 2402 informs you that the database was successfully exported, as shown in FIGURE 20.



DFS Replication Number of events: 43

Level	Event ID	Source	Date and Time	Task Category
Information	2002	DFSR	5/14/2020 4:33:00 PM	None
Information	2402	DFSR	5/14/2020 4:33:00 PM	None
Information	2410	DFSR	5/14/2020 4:32:59 PM	None
Information	2406	DFSR	5/14/2020 4:32:58 PM	None
Information	4112	DFSR	5/14/2020 3:45:03 PM	None

Event 2402, DFSR

The DFS Replication service successfully exported a cloned database.

Additional Information:
Volume: R:\

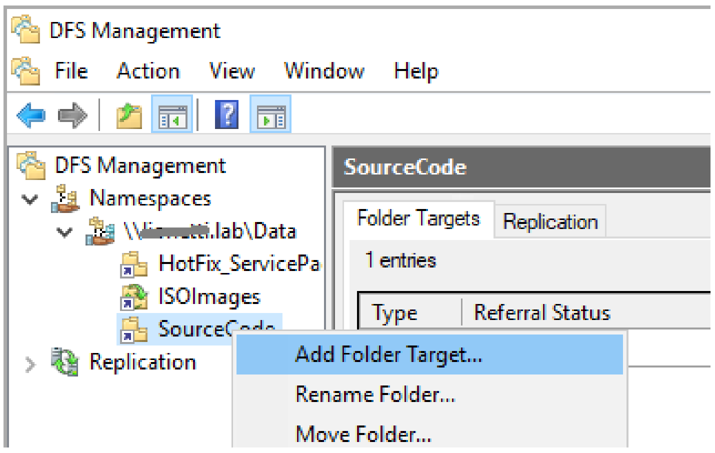
Log Name:	DFS Replication		
Source:	DFSR	Logged:	5/14/2020 4:33:00 PM
Event ID:	2402	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	DFSR1.[redacted].lab
OpCode:			
More Information:	Event Log Online Help		

FIGURE 20. DFS example that database was successfully exported

– If array-to-array replication is used, wait until both copies are in sync, or until the last update from an asynch replication is newer than the time at which the database successfully exports according to the Event Log.

– If an array snapshot or clone is used, take the snapshot at this time, and expose the new volume to the DFSR2 server. Once exposed to this server, you should be able to bring the drive online, as well as assigning it an appropriate drive letter. Next, re-share DFSR2 `F:\SourceCode` as a share so that it can be added as a folder target for the `\namespace\Data\SourceCode` dataset.

FIGURE 21 shows how to add that snapshot folder to DFSR.


DFS Management

File Action View Window Help

DFS Management
- Namespaces
  - \\[redacted].lab\Data
    - HotFix_ServicePa
    - ISOImages
    - SourceCode
- Replication

SourceCode

Folder Targets | Replication

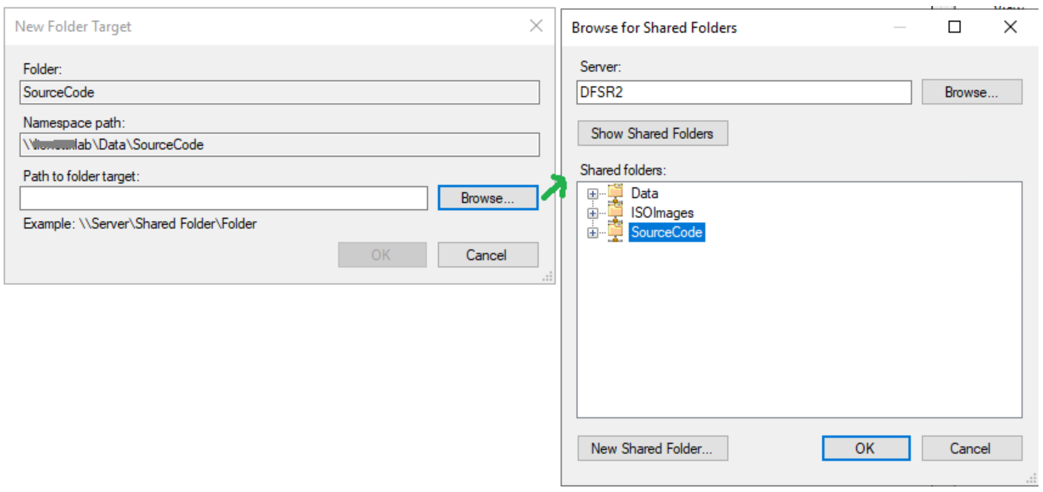
1 entries

Type	Referral Status

Add Folder Target...
Rename Folder...
Move Folder...

FIGURE 21. Adding the folder into DFSR target site.

FIGURE 22 illustrates the required settings to complete the operation


New Folder Target

Folder: SourceCode

Namespace path: \\[redacted]lab\Data\SourceCode

Path to folder target:

Example: \\Server\Shared Folder\Folder

Browse... OK Cancel

Browse for Shared Folders

Server: DFSR2

Browse... Show Shared Folders

Shared folders:
- Data
- ISOImages
- SourceCode

New Shared Folder... OK Cancel

FIGURE 22. Settings required to complete the folder add process

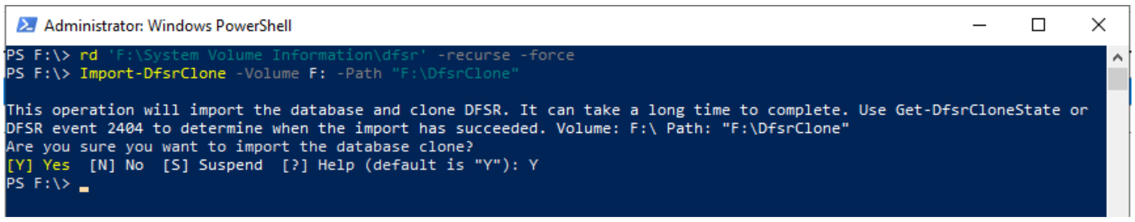
6. When prompted to create a replication group, answer "no."
7. At this point, import the "DfsrClone" that was previously exported from DFSR1.

   From the DFSR2 server, issue the following PowerShell command:

```
PS:> RD 'E:\System Volume Information/DFSR' -recurse -force
PS:> Import-DfsrClone -Volume E: -path E:\DfsrClone
```

NOTE
This command will fail because there is a hidden directory named "SystemVolumeInformation," which contains the database from the DFSR1 server and it must be removed to add this database.

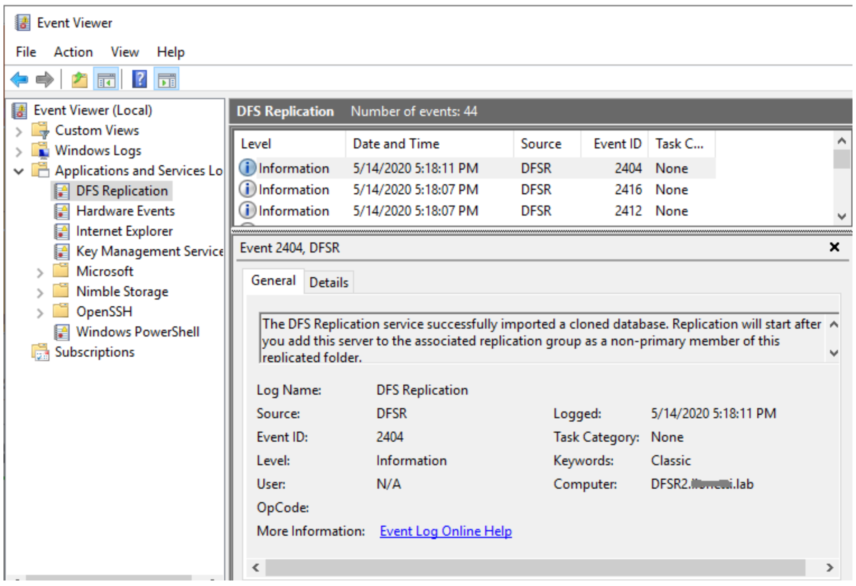
FIGURE 23 shows the PowerShell command in action that will import the DFSR clone.



```
PS F:\> rd 'F:\System Volume Information\dfsr' -recurse -force
PS F:\> Import-DfsrClone -Volume F: -Path "F:\DfsrClone"

This operation will import the database and clone DFSR. It can take a long time to complete. Use Get-DfsrCloneState or
DFSR event 2404 to determine when the import has succeeded. Volume: F:\ Path: "F:\DfsrClone"
Are you sure you want to import the database clone?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y
PS F:\> _
```

FIGURE 23. PowerShell command to import the database clone

8. Wait until the import has completed, and look for DFS Replication Event ID 2404, as shown in FIGURE 24.



DFS Replication Number of events: 44

Level	Date and Time	Source	Event ID	Task C...
Information	5/14/2020 5:18:11 PM	DFSR	2404	None
Information	5/14/2020 5:18:07 PM	DFSR	2416	None
Information	5/14/2020 5:18:07 PM	DFSR	2412	None

Event 2404, DFSR

The DFS Replication service successfully imported a cloned database. Replication will start after you add this server to the associated replication group as a non-primary member of this replicated folder.

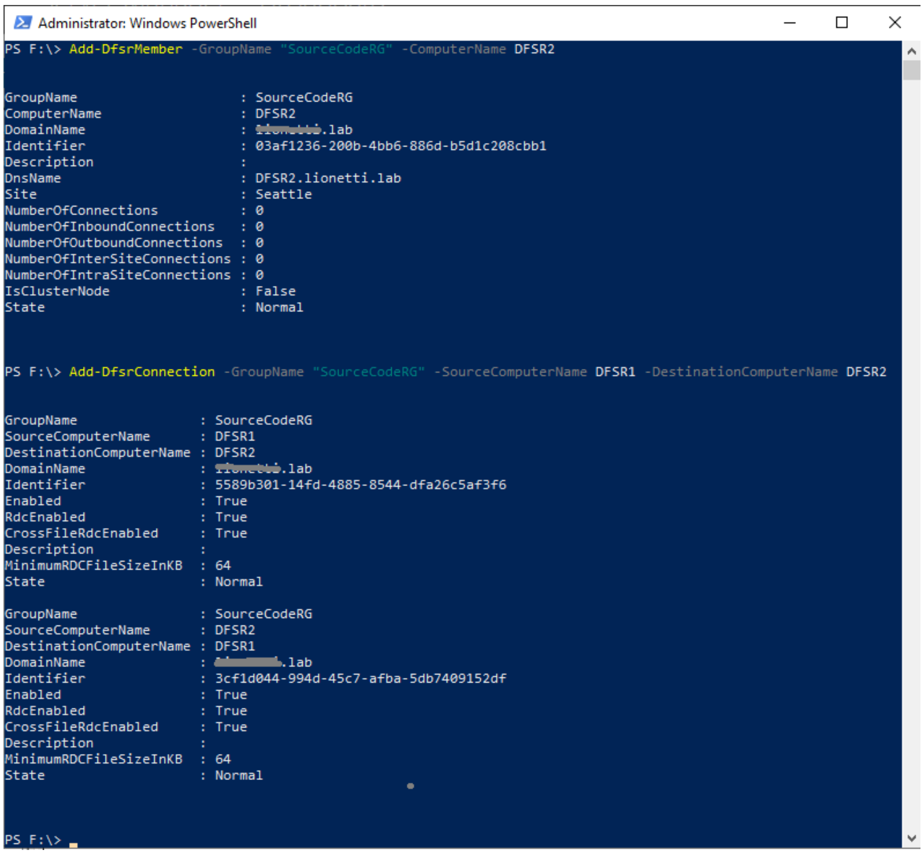
Log Name:	DFS Replication		
Source:	DFSR	Logged:	5/14/2020 5:18:11 PM
Event ID:	2404	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	DFSR2.[redacted].lab
OpCode:			
More Information:	Event Log Online Help		

FIGURE 24. Example of DFS Replication Event ID 2404

9. Add the DFSR2 server as the downstream replication group member for this DFSR target folder. Use the following two sets of four commands to accomplish this task. Here are the first two commands:

```
PS:> Add-DfsrMember -GroupName "SourceCodeRG" -ComputerName DFSR2

PS:> Add-DfsrConnection -GroupName "SourceCodeRG" -SourceComputerName DFSR1 -DestinationComputerName
DFSR2
```

FIGURE 25 shows the PowerShell output for the first two commands illustrated above.



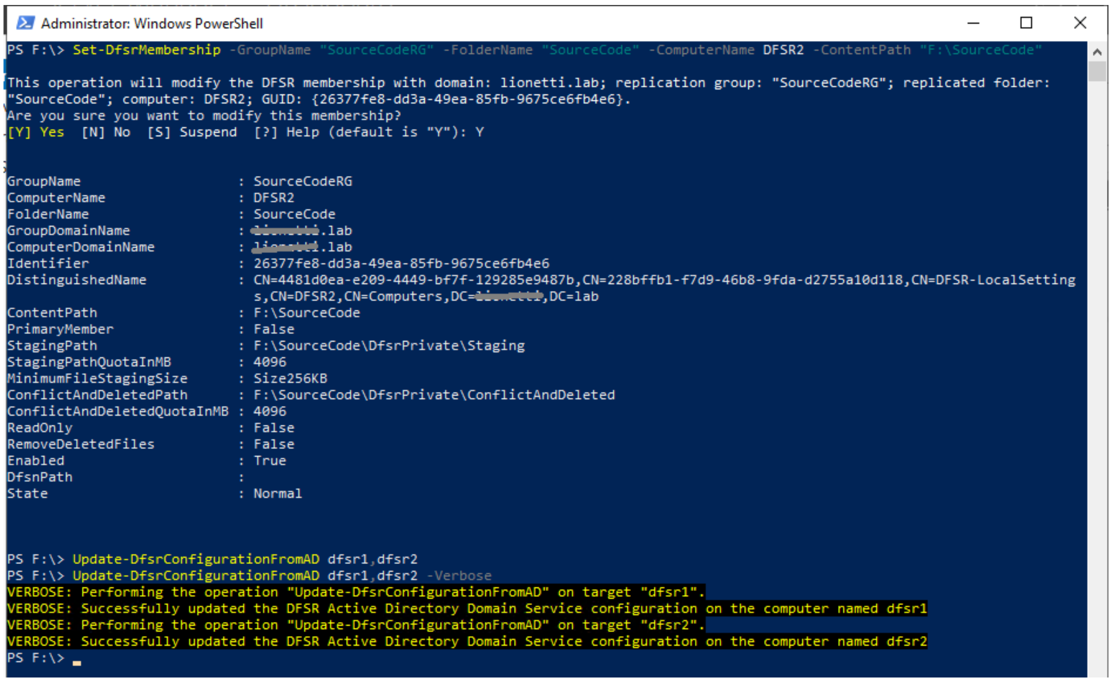
```
PS F:\> Add-DfsrMember -GroupName "SourceCodeRG" -ComputerName DFSR2


GroupName                     : SourceCodeRG
ComputerName                  : DFSR2
DomainName                    : [redacted].lab
Identifier                    : 03af1236-200b-4bb6-886d-b5d1c208cbb1
Description                   :
DnsName                       : DFSR2.lionetti.lab
Site                          : Seattle
NumberOfConnections           : 0
NumberOfInboundConnections    : 0
NumberOfOutboundConnections   : 0
NumberOfInterSiteConnections  : 0
NumberOfIntraSiteConnections  : 0
IsClusterNode                 : False
State                         : Normal



PS F:\> Add-DfsrConnection -GroupName "SourceCodeRG" -SourceComputerName DFSR1 -DestinationComputerName DFSR2


GroupName                : SourceCodeRG
SourceComputerName       : DFSR1
DestinationComputerName  : DFSR2
DomainName               : [redacted].lab
Identifier               : 5589b301-14fd-4885-8544-dfa26c5af3f6
Enabled                  : True
RdcEnabled               : True
CrossFileRdcEnabled      : True
Description              :
MinimumRDCFileSizeInKB   : 64
State                    : Normal

GroupName                : SourceCodeRG
SourceComputerName       : DFSR2
DestinationComputerName  : DFSR1
DomainName               : [redacted].lab
Identifier               : 3cf1d044-994d-45c7-afba-5db7409152df
Enabled                  : True
RdcEnabled               : True
CrossFileRdcEnabled      : True
Description              :
MinimumRDCFileSizeInKB   : 64
State                    : Normal



PS F:\> _
```

FIGURE 25. PowerShell commands output of adding the DFSR member and Connection

And here are the last two commands:

```
PS:> Set-DfsrMembership -GroupName " SourceCodeRG " -FolderName "SourceCode" -ComputerName DFSR2
-ContentPath "F:\SourceCode"

PS:> Update-DfsrConfigurationFromAD DFSR1,DFSR2
```

FIGURE 26 shows the PowerShell output for the last two commands shown above.



```
PS F:\> Set-DfsrMembership -GroupName "SourceCodeRG" -FolderName "SourceCode" -ComputerName DFSR2 -ContentPath "F:\SourceCode"

This operation will modify the DFSR membership with domain: lionetti.lab; replication group: "SourceCodeRG"; replicated folder:
"SourceCode"; computer: DFSR2; GUID: {26377fe8-dd3a-49ea-85fb-9675ce6fb4e6}.
Are you sure you want to modify this membership?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y


GroupName                    : SourceCodeRG
ComputerName                 : DFSR2
FolderName                   : SourceCode
GroupDomainName              : [redacted].lab
ComputerDomainName           : [redacted].lab
Identifier                   : 26377fe8-dd3a-49ea-85fb-9675ce6fb4e6
DistinguishedName            : CN=4481d0ea-e209-4449-bf7f-129285e9487b,CN=228bffb1-f7d9-46b8-9fda-d2755a10d118,CN=DFSR-LocalSetting
                               s,CN=DFSR2,CN=Computers,DC=[redacted],DC=lab
ContentPath                  : F:\SourceCode
PrimaryMember                : False
StagingPath                  : F:\SourceCode\DfsrPrivate\Staging
StagingPathQuotaInMB         : 4096
MinimumFileStagingSize       : Size256KB
ConflictAndDeletedPath       : F:\SourceCode\DfsrPrivate\ConflictAndDeleted
ConflictAndDeletedQuotaInMB : 4096
ReadOnly                     : False
RemoveDeletedFiles           : False
Enabled                      : True
DfsnPath                     :
State                        : Normal



PS F:\> Update-DfsrConfigurationFromAD dfsr1,dfsr2
PS F:\> Update-DfsrConfigurationFromAD dfsr1,dfsr2 -Verbose
VERBOSE: Performing the operation "Update-DfsrConfigurationFromAD" on target "dfsr1".
VERBOSE: Successfully updated the DFSR Active Directory Domain Service configuration on the computer named dfsr1
VERBOSE: Performing the operation "Update-DfsrConfigurationFromAD" on target "dfsr2".
VERBOSE: Successfully updated the DFSR Active Directory Domain Service configuration on the computer named dfsr2
PS F:\> _
```

FIGURE 26. PowerShell output for the “Set-DfsrMembership” and “Update-DfsrConfigurationFromAD” commands

Advanced DFSR – The case for hardware-based snapshots

The weakness of DFSR is that bad actions, such as inadvertent or malicious deletions on one site, can be so easily and automatically replicated or honored on the other sites in the replication group. To revert a file system back to a previous version takes great care because the replication service will attempt to overwrite those older versions of the files from the alternate sites as soon as replication is re-enabled. In this case, perform the following procedure:

1. Turn off replication between sites (on that share specifically).
2. Choose the master site, and restore the dataset to its proper snapshot
3. Choose the alternate sites and restore their datasets to a snapshot prior to the above snapshot, because you want the authoritative copy to be from the snapshot in step 2.
4. Re-create the replication group between the sites and start replication.

This process is only needed for a full backup. If you are also using the Microsoft embedded VSS snapshot (recommended) you can restore a single file by reverting to its previous version. If you revert a file to its previous version, however, you might need to take an extra step to update its timestamp code to be sure that the file is not overwritten on the next compare with the alternate sites. This can be done via the following simple PowerShell command:

```
PS:>  (Get-ChildItem .\MyFile.txt).LastWriteTime= get-date
```

Alternately, you can also choose to restore single files or directories from a hardware based VSS snapshot. This can be done on either HPE Nimble Storage or HPE Primera using the PowerShell Toolkits, or manually via their respective GUIs. The steps for this process are as follows:

1. Identify the snapshot with the date code that contains the files or directory to restore.
2. Mount the snapshot to the host using a mount point or other free drive letter.
3. For HPE Nimble Storage use the following PowerShell command:

   ```
   PS:>Invoke-NimCloneVolume -Snapshot ? -accessDriveLetter
   ```

4. For HPE 3PAR and HPE Primera storage, use the following PowerShell command:

```
PS:> New-VvCopy | Set-VvCopy
```

5. After the point-in-time copy is mounted, find the files or folders you want to copy back to the data source.
6. If the file or folder is to be overwritten, update the modified data as shown above to make sure the file is not overwritten at a later time.
7. If the file or folder is to be restored, but not overwritten, simply copy the data into place.

SUMMARY

DFS and DFSR are only two of the possible options to solve real-world file serving and replication needs. Other options include BranchCacheV2 and Azure File Sync, as well as a number of third-party products and the capability of customers to create their own unique solution. The DFS and DFSR options have a number of weaknesses regarding long-term retention and survivability and initial replication. These limitations happen to match extremely well with strengths in both HPE Nimble Storage and HPE Primera/HPE 3PAR storage snapshot and replication technologies. These two technologies—DFSR and HPE SAN storage—are an example where they are better together, and offer additive value when deployed together.

RESOURCES AND ADDITIONAL LINKS

HPE Nimble Storage
hpe.com/us/en/storage/nimble.html

HPE Primera Storage
hpe.com/us/en/storage/hpe-primera.html

HPE 3PAR StoreServ Storage
hpe.com/us/en/storage/3par.html

Microsoft DFSR Overview
docs.microsoft.com/en-us/previous-versions/windows/desktop/dfsr/dfsr-overview

LEARN MORE AT

hpe.com/Storage/Microsoft

Make the right purchase decision.
Contact our presales specialists.

Chat **Email** **Call**

Get updates

a00110612ENW, February 2021, Rev 1