

# Deploying and Managing Exchange Server 2013 High Availability



 **exchangeserverpro.com**

© Copyright 2015 LockLAN Systems Pty Ltd

The right of LockLAN Systems Pty Ltd to be identified as author and copyright owner of this work is asserted by LockLAN Systems Pty Ltd in accordance with Australian copyright laws as determined by the Australian Copyright Council.

Copyright extends to any and all countries in which this publication is purchased and/or viewed and/or read.

All rights reserved. No part of this publication may be reproduced or transmitted, in any form by any means without the prior written permission of the author, nor be otherwise circulated in any form of binding or cover other than that in which it is published and without a similar condition being imposed on the subsequent purchaser.

The purchaser of this publication indemnifies Paul Cunningham and LockLAN Systems Pty Ltd and its directors, officers, employees and agents from and against all losses, claims, damages and liabilities which arise out of any use of this publication and/or any application of its content.

To buy a copy of this ebook visit <http://exchangeserverpro.com/ebooks>.

If you find an error in this ebook we'd love to fix it. Please let us know at  
<http://exchangeserverpro.com/contact>.

# About the Authors

## Paul Cunningham



Paul Cunningham is an Exchange Server MVP and Systems Consultant living in Brisbane, Australia. Paul is the founder of [ExchangeServerPro.com](http://ExchangeServerPro.com), a leading website in the Exchange Server community. He also holds several Microsoft certifications including for Exchange Server 2007, 2010 and 2013.

Connect with Paul on [Twitter](#) (@ExchServerPro) and [LinkedIn](#).

## Michael Van Horenbeeck



Michael Van Horenbeeck is the Director of Product Research for ENow Software, an Exchange Server MVP, and a Microsoft Certified Solutions Master from Belgium who specializes in Office 365 and Hybrid deployments.

Connect with Michael on [Twitter](#) (@mvanhorenbeeck) and [LinkedIn](#).

## Steve Goodman



Steve Goodman is Head of Unified Communications for Content and Code, one of the United Kingdom's top Microsoft partners, and an Exchange Server MVP. Steve is an expert in complex, large scale Exchange Server deployments as well as Office 365.

Connect with Steve on [Twitter](#) (@stevegoodman) and [LinkedIn](#).

*The authors would like to extend a special thanks to Tony Redmond, Tony Murray, Scott Schnoll and Tim Heeney for their assistance with this ebook.*

# Table of Contents

Introduction .....	1
What is High Availability? .....	2
What Does High Availability Mean for Exchange Server 2013? .....	3
There's More to High Availability Than Just Exchange Server 2013 .....	3
Why Do We Want High Availability for Exchange Server 2013?.....	4
Why Don't We Want High Availability for Exchange Server 2013? .....	4
Alternatives to Exchange Server 2013 High Availability .....	5
What about Virtualization?.....	5
How to Use This Guide.....	6
Conventions Used in This Guide .....	6
Build Your Own Test Lab .....	7
Client Access server High Availability .....	8
Overview of Client Access Server Role.....	8
New Client Access Server Architecture.....	8
The Demise of RPC.....	9
High Availability .....	10
Load Balancing Concepts .....	10
Persistence (And Why it isn't Important in Exchange Server 2013) .....	11
Load Balancing Mechanisms .....	12
Health Checking .....	12
SSL Bridging and SSL Offloading.....	15
Layer 4 / Layer 7 Load Balancing.....	16
SSL Certificate Requirements.....	17
Namespace Planning.....	17
Split-DNS .....	20
POP, IMAP & SMTP .....	22
Deciding on a Load Balancing Model.....	23
DNS Round Robin.....	23
Layer 4 Load Balancing with a Single Namespace .....	27

Layer 4 Load Balancing with Multiple Namespaces.....	31
Layer 7 Load Balancing.....	33
What About Multi-Site Load Balancing?.....	36
DNS round-robin .....	36
Geo DNS .....	37
Regional Namespaces .....	38
Configuring Client Access Servers .....	39
Configuring DNS Records .....	39
Configuring Virtual Directories .....	41
Configuring an SSL Certificate .....	42
Client Access Server Summary.....	46
Mailbox Server High Availability .....	47
Overview of Exchange Server 2013 Database Availability Groups .....	47
DAG Members.....	49
Incremental Deployment .....	49
Quorum.....	50
Dynamic Quorum .....	51
File Share Witness.....	52
Active and Passive Database Copies.....	53
Continuous Replication.....	54
Lagged Database Copies .....	55
Active Manager.....	55
Switchovers and Failovers.....	55
Activation Preference .....	56
Best Copy and Server Selection .....	56
High Availability vs Site Resilience .....	57
Datacenter Activation Coordination Mode .....	58
Related Transport Features .....	64
Circular Logging.....	64
DAG Server and Storage Design.....	65
Server Sizing Calculator.....	66
Multiple Databases per Volume .....	66

RAID vs JBOD .....	67
Disk Layout Examples.....	68
Configuring Storage for Exchange Server 2013 .....	70
Create New Volumes .....	71
Create Root Directories.....	72
Configure the Number of Databases per Volume .....	73
Mount the Volume Folders .....	73
Create the Database Folders.....	75
Create the Mount Points for Databases .....	76
Create the Database Directory Structure .....	77
DAG Network Design .....	78
DAG Network Types.....	78
Dedicated Replication Networks .....	79
DAG Network Auto-Configuration .....	79
Recommendations for DAG Networks.....	79
Deploying and Managing a Database Availability Group.....	80
DAG Pre-Requisites .....	80
Pre-Staging the Cluster Name Object .....	81
Preparing the File Share Witness .....	84
Creating a New Database Availability Group.....	85
Managing Database Availability Group Members .....	88
Managing DAG Networks.....	91
Managing Database Copies.....	94
Managing Database Switchovers.....	103
Database Failovers .....	104
Configuring DAC Mode .....	105
Extending DAGs to Multiple Sites .....	105
One DAG or Two for Site Resilience? .....	106
Placing a File Share Witness in a Third Site.....	106
Demonstration: Site Failover for Disaster Recovery.....	107
Lagged Database Copies .....	112
Configuring Lagged Database Copies.....	112

Using Lagged Database Copies in Recovery Scenarios .....	113
After Lagged Copy Activation.....	120
A Deeper Look at DAG Features .....	120
Best Copy and Server Selection in Action .....	121
Dynamic Quorum in Action.....	124
Auto-Reseed in Action .....	126
Mailbox Server Summary.....	130
Transport High Availability.....	131
Exchange Server 2013 Transport Architecture .....	131
Client Access Server Transport Services .....	132
Mailbox Server Transport Services .....	132
Routing Destinations.....	133
Delivery Groups.....	133
Front End Transport Service Email Routing .....	134
Mailbox Transport Service Email Routing .....	135
High Availability for Transport Services .....	137
Inbound Internet Email .....	137
Outbound Internet Email .....	139
Internal SMTP Senders.....	140
High Availability Features of Mailbox Transport Services.....	141
Shadow Redundancy.....	141
Safety Net.....	143
How Safety Net and Shadow Redundancy Work Together .....	144
Preserving Safety Net and Shadow Redundancy .....	144
Edge Transport Server High Availability.....	145
Single Datacenter and Internet Connection .....	146
Multiple Datacenters and Internet Connections .....	148
Effect of Connector Costs on Email Routing .....	150
Transport Summary .....	152
High Availability for Unified Messaging .....	153
Unified Messaging Concepts.....	154
UM Dial Plans .....	154

UM Mailbox Policies .....	154
IP Gateways.....	154
UM Hunt Groups.....	155
UM Auto Attendants.....	155
Connecting to Phone Systems .....	155
Unified Messaging Components .....	156
The UM Call Router Service .....	156
The UM Service .....	156
Certificate Requirements .....	157
Unified Messaging High Availability.....	157
DAG Design Implications.....	158
Network Implications.....	158
Sizing Implications.....	158
Unified Messaging Summary .....	159
Managing and Monitoring High Availability .....	160
Managed Availability.....	160
Health Manager Service.....	161
Health Manager Worker process.....	162
File Locations.....	162
HealthSets .....	164
Probes .....	164
Monitors.....	167
Monitoring Overrides.....	169
Responders .....	171
Throttling .....	172
Server Component States .....	172
Performing Server Maintenance.....	177
Putting a Server into Maintenance Mode.....	178
Verifying That a Server Has Successfully Been Placed into Maintenance Mode .....	180
Taking a Server Out of Maintenance Mode .....	182
Server Recovery .....	184
Storage locations .....	184

Backup Considerations.....	184
Backup Software .....	185
Recovery Databases.....	186
Recovering Servers with the /m:RecoverServer Switch .....	186
Managing and Monitoring Summary .....	192
High Availability for Hybrid Deployments.....	193
Hybrid Concepts.....	194
Secure Mail flow .....	194
Directory Synchronization.....	194
Remote Mailbox Moves .....	195
Rich Coexistence (Exchange Federation) .....	195
Bringing it all Together.....	197
Why is High Availability Important in a Hybrid Deployment? .....	197
Other High Availability Components in a Hybrid Deployment .....	199
High Availability for Directory Synchronization .....	199
High Availability for Client Access Servers in a Hybrid Deployment.....	200
High Availability for Mailbox Servers in a Hybrid Deployment .....	201
Regional Considerations for Mailbox Migrations .....	202
High Availability for Transport in a Hybrid Deployment.....	204
Authentication Using Active Directory Federation Services .....	205
Considerations for Publishing AD FS onto the Internet .....	207
Hybrid AD FS Deployments .....	208
Authentication with Password Synchronization.....	209
Hybrid Deployment Summary.....	210
Appendix A – Lab Guide .....	211
Preparing to Build a Test Lab Environment .....	212
Installing Hyper-V on Windows 8.1.....	212
Downloading Software.....	213
What Else Do You Need? .....	213
Building Virtual Machines in Hyper-V .....	213
Configuring a Virtual Switch.....	214
Creating a Virtual Hard Disk to Use as a Base Image .....	215

Creating a Virtual Machine to Use as a Base Image .....	217
Installing the Domain Controller.....	225
Install Certificate Services .....	231
Install Exchange Server 2013 .....	235
Adding Some Realism to the Test Lab.....	242
Creating Test Users and Mailboxes.....	242
Simulating Email Traffic .....	243
Configuring Backups.....	243
Installing a Client Machine.....	243
Implementing Unified Messaging .....	244
Certificate Configuration.....	244
Creating the Unified Messaging Dial Plan.....	246
Configuring Exchange for Lync Integration.....	249
Enable Mailboxes for Unified Messaging.....	251
Testing Unified Messaging in Lync and Outlook Web App .....	253
Lab Guide Summary .....	254

# Introduction

Welcome to Deploying and Managing High Availability for Exchange Server 2013.

High availability has become an essential part of the email services in many organizations around the world, which means that understanding how to deploy and maintain a highly available Exchange Server environment is a critical skill for Exchange Server administrators.

Microsoft Exchange Server 2013 builds upon the high availability features of previous versions of Exchange, as well as introducing many architectural changes and new features.

In this guide we'll go through these changes and features, explain how they work, and demonstrate how they can be used in the real world when you are working with Exchange Server 2013 for your customers.

The chapters in this guide cover:

- Client Access server high availability, including load balancing, namespace planning, and SSL certificates.
- Mailbox server high availability, including deploying and configuring Database Availability Groups, multi-site considerations, and using features such as lagged database copies.
- Transport high availability, including the features of Exchange Server 2013 that protect email in transit, and Edge Transport servers.
- Unified Messaging high availability, including how the architectural changes in Exchange Server 2013 impact UM.
- Managing and monitoring high availability, including backups, recovery scenarios, and Managed Availability.
- Hybrid configuration high availability, including the considerations for organizations integrating with Office 365.

# What is High Availability?

In the world of Information Technology (IT) the term “high availability” generally refers to a system or service that has been designed and implemented to meet a desired level of performance and availability.

We’re sure you are already familiar with the term “high availability”, but let’s discuss it here first to lay the foundation.

A service is “available” when it is accessible by end users, and is successfully performing the tasks that the service is designed to fulfil. This is also referred to as “uptime”.

A service is “unavailable” when it is not accessible by the end users. This is also referred to as “downtime”. Depending on the organization “downtime” may or may not include scheduled downtime for systems.

High availability doesn’t necessarily mean “always available”, but it does tend to mean a very high percentage of uptime during the hours of the day that the service is required, such as 98%, 99.9%, or even the “five nines” target of 99.999%.

Availability is usually measured across an entire year. Here are some common availability targets<sup>1</sup>, along with what they mean in terms of weekly, monthly, and yearly downtime.

Availability Target %	Yearly Downtime	Monthly Downtime	Weekly Downtime
90%	36.5 days	72 hours	16.8 hours
95%	18.25 days	36 hours	8.4 hours
99%	3.65 days	7.2 hours	1.68 hours
99.5%	1.83 days	3.6 hours	50.5 minutes
99.9%	8.76 hours	43.8 minutes	10.1 minutes
99.99%	52.56 minutes	4.32 minutes	1.01 minutes
99.999%	5.26 minutes	25.9 seconds	6.05 seconds

As you can see a high availability target % leaves very little room for unplanned downtime.

Availability targets should be driven by a customer’s genuine business needs, although often they are just a number out of thin air with no real business justification behind it.

For example, a business might determine that they can afford only 1 hour of downtime for their email service in a month. So their availability target could be defined as 99.9%, which means about 43 minutes of downtime per month.

---

<sup>1</sup> [http://en.wikipedia.org/wiki/High\\_availability#Percentage\\_calculation](http://en.wikipedia.org/wiki/High_availability#Percentage_calculation)

Availability targets such as those become the basis for the design of the service, as well as the operational procedures surrounding it.

For example, servers need to be maintained with security patches and other updates, which means some planned downtime of components of the service will be necessary at regular intervals.

So the email service would require a technical design that allows some parts to be unavailable at different times without interrupting the availability of the service, and operational procedures to be followed when performing planned maintenance.

At the same time, the business wants to avoid unplanned downtime due to hardware failure or human error. Which means the service needs to be made up of multiple, redundant components that can withstand different points of failure. Also, adequate controls need to be in place to ensure that an unskilled or even malicious person doesn't cause a service interruption.

## What Does High Availability Mean for Exchange Server 2013?

Although Microsoft Exchange Server 2013 is a single product it is made up of many different components, such as:

- Client access – the protocols that allow mailbox access via Outlook, Outlook Web App (OWA), ActiveSync (mobile devices), and other clients.
- Transport – SMTP communications and mail flow.
- Mailbox – the databases hosting mailboxes.
- Unified Messaging – the telephony integration services.

When we talk about high availability for Exchange Server 2013 we need to be specific about the availability targets of those different components.

For example, an education facility might require 99.9% availability for staff email access, but only 90% availability for student email access.

Or as another example, a business running a critical transaction system that uses SMTP might require 99.999% availability for Transport, but only 99% for mailboxes and client access.

## There's More to High Availability Than Just Exchange Server 2013

As well as being a complex product itself, Exchange Server 2013 relies on a series of dependencies for its availability.

For example:

- Active Directory
- Domain Name System (DNS)
- Load balancers
- Network connectivity (WAN, LAN, Internet)
- Windows Server operating system
- Server hardware
- Virtualization
- Disk storage
- Data centre power

In the context of a highly available Exchange Server 2013 implementation all of those dependencies need to have similar levels of availability for the service as a whole to function correctly.

For the purposes of this guide we will discuss a few of those dependencies in more detail, however we will not cover every possible dependency of an Exchange Server 2013 environment.

The full list of dependencies is something that will depend on the specific environment, so we do encourage you to do a thorough analysis of your own environment when you are considering high availability for Exchange Server 2013.

## Why Do We Want High Availability for Exchange Server 2013?

The communications and collaboration services provided by Exchange Server 2013 are some of the most highly visible and heavily utilized services in most organizations.

This puts any Exchange Server availability issues high on the list of pain points for an IT department. The last thing IT staff want to hear from end users are complaints that “email is down again”.

There needs to be a valid business reason behind it, such as a measurable cost (e.g. sales lost, or data re-processing required) of each hour of service downtime. The IT department’s discomfort is not a strong justification for implementing high availability, but reducing the cost of IT personnel may well be a valid reason for a highly available solution.

Ultimately we need to ensure that any high availability solution implemented for Exchange Server realises an overall benefit. There is no point investing in a very complex and expensive solution if it only returns a minor financial benefit to the business.

## Why Don’t We Want High Availability for Exchange Server 2013?

The reason we need a measurable cost for downtime to justify high availability is because implementing high availability comes at a price; both in terms of monetary cost and complexity.

Highly available Exchange Server 2013 systems generally mean multiple servers, load balancers, redundant network paths, and in some cases even multiple data centres.

All of those additional servers and components obviously costs more money than a single server and simpler network would.

It also means the environment is more complex to manage and maintain, which can increase training costs and salaries for IT staff to ensure that appropriately skilled operators are keeping the Exchange Server environment running.

After all, there's a big difference in skills required to manage a single Exchange server compared to managing a multi-site Database Availability Group.

## Alternatives to Exchange Server 2013 High Availability

If the costs and complexity of a highly available Exchange Server 2013 implementation can't be justified there are other options a business can consider.

One option is to simply accept the risk of more frequent downtime by running on a single Exchange server. The risk of lengthy downtime can often be mitigated by purchasing good quality server hardware, with a support contract that ensures fast resolution of hardware failures, and with good backup and recovery processes for any situations where data loss may have occurred.

Another option is to move the email service into the cloud using Office 365, Microsoft's cloud service that includes Exchange Online. For many organizations the per-user licensing of Office 365 makes it an attractive option as an affordable, yet highly available email service.

## What about Virtualization?

You might be wondering if virtualization can be used to improve the availability of Exchange Server 2013. The answer of course, is that it depends.

Different virtualization hypervisors offer features that can improve the availability of an Exchange Server, such as the ability to move the virtual machine between different host machines, and the ability to automatically restart a virtual machine that has crashed.

However these features require an investment in virtualization infrastructure that in itself needs to be implemented the correct way so that the guest virtual machines are less prone to certain failure scenarios.

That virtualization infrastructure may add cost and complexity to your IT environment, which as discussed previously may not be desirable.

On the other hand, a business that has already invested in virtualization may perform all of the necessary analysis and find that it would be more cost effective to host their Exchange Server 2013

server on that existing infrastructure. Alternatively, they may decide that even when it is found to be more cost effective to run Exchange Server 2013 on dedicated servers, other factors such as political pressure mean that virtualization is chosen anyway.

But at the end of the day virtualizing a single Exchange server only prevents some failure scenarios (eg, server hardware failure) from causing downtime, but still leaves the server susceptible to other downtime events such as operating system malfunction, administrative error, application component failure, or database corruption.

For example, a hypervisor-based high availability model may only be able to detect and respond to server failures, leaving other application component failures such as an unhealthy ActiveSync protocol on a Client Access server undetected and causing problems for end users. Whereas a solution designed around the high availability features of Exchange Server 2013 will detect such issues and either resolve them automatically or work around them in a way that avoids a service interruption for end users.

## How to Use This Guide

Deploying and Managing High Availability for Exchange Server 2013 has been written to provide value to you in a variety of ways.

You can read this guide from front to back to learn about Exchange Server 2013 high availability, or you can use it as a reference that you dip into for specific information from time to time.

Because many of the chapters include step by step instructions and demonstrations, you can also use this guide for training purposes and work along with it in your own test lab environment.

## Conventions Used in This Guide

Throughout this guide some formatting is used to present specific information.

PowerShell cmdlets are presented in text boxes like this.

```
[PS] C:\>Get-ExchangeServer
Name          Site          ServerRole Edition AdminDisplayVersion
----          ---          -----   -----
E15MB1        exchange2013demo.... Mailbox,... Enterprise Version 15.0 (Bu...
```

Some long PowerShell commands don't fit on a single line and the text will wrap to multiple lines instead. Where multiple commands are shown in a single example each separate command line is preceded by the PowerShell prompt, eg [PS] C:\>.

For example, this is a single PowerShell command that has wrapped to multiple lines.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | where {$_.ContentIndexState -eq "Failed"} | Update-MailboxDatabaseCopy -CatalogOnly
```

Whereas this is two PowerShell commands run one after the other.

```
[PS] C:\>$failedindexes = Get-MailboxDatabaseCopyStatus * | where {$_.ContentIndexState -eq "Failed"}  
[PS] C:\>$failedindexes | Update-MailboxDatabaseCopy -CatalogOnly
```

Some topics might have an extra note or real world example in them. These will be presented in colour text boxes like this.

**Note:** An extra note about a section in the book will appear like this. It may include references to extra material on the internet such as whitepapers or knowledgebase articles.

**Real World:** An example from a real world deployment will appear like this. The actual customer name might not be included though for confidentiality reasons

**Warning:** Warnings or other cautionary notes will appear like this. Try not to ignore these, the lessons were often learned the hard way and we'd hate to see you suffer the same pain.

## Build Your Own Test Lab

If you would like to build your own test lab environment to work along with this guide then please refer to Appendix A, which includes some suggestions on hardware and virtualization approaches to test lab environments, as well as demonstrating step by step how to set up a test lab using a computer running Windows 8.1 and Hyper-V.

# Client Access server

# High Availability

The Client Access server (CAS) is the primary entry-point for all client connections to Exchange Server 2013. This means that it is important when you are designing for or configuring high availability for your environment that you also take into account the Client Access servers and the supporting network infrastructure, including the different access paths to your CAS infrastructure.

## Overview of Client Access Server Role

Compared to earlier versions of Exchange, the Client Access server role has changed quite a bit in Exchange 2013. These changes have made it easier to design for and configure high availability for this server role.

Let's first have a look at how the Client Access server role has evolved, and what its current role is in an Exchange Server 2013 environment.

### New Client Access Server Architecture

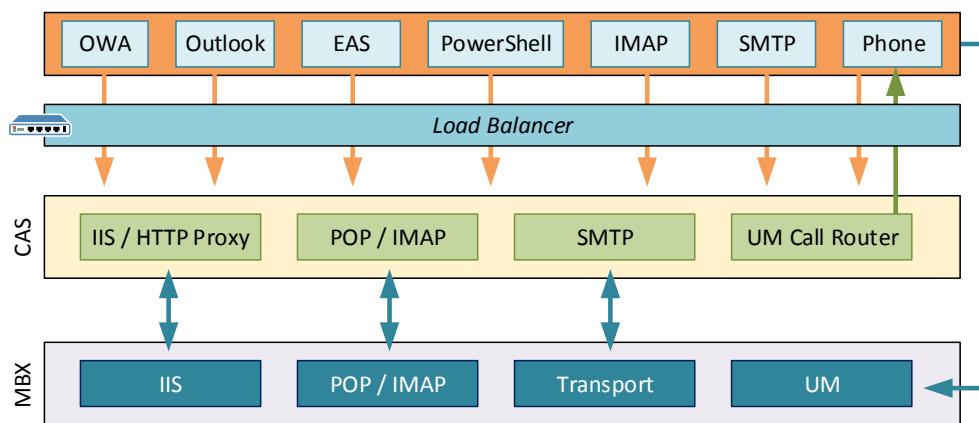
In the new Exchange 2013 CAS architecture we can identify three main areas for which the CAS is responsible:

- Responsible for Client Connectivity Protocols (HTTP, POP & IMAP)
- Front-end Transport Service (SMTP)
- Unified Messaging (UM) Call Router

Within these areas, the CAS will authenticate traffic (if applicable) and forward it to the appropriate Mailbox server. For client connectivity protocols, traffic will always be forwarded to the Mailbox server hosting the active copy of the mailbox database in which the user's mailbox is located.

Routing of SMTP traffic follows a totally different logic and is explained in the Transport chapter of this guide.

Unlike Exchange Server 2010 the Client Access server in Exchange Server 2013 is no longer responsible for rendering any data. The Exchange Server 2013 CAS is only a stateless proxy. At no point – not even with SMTP traffic – will data be stored on the server. All it does is accept incoming connections and forward them to the appropriate destination.



As you can see in the diagram above, the CAS acts as a proxy for all protocols request except for Unified Messaging-related traffic. Because of the real-time nature of UM traffic, acting as a man-in-the-middle could cause unnecessary delays.

Instead, whenever UM traffic hits the CAS, it will look up which Mailbox server is currently responsible for the user's mailbox and then issue a SIP REDIRECT message for the specific server. As a result, subsequent connections (within that same SIP session) will now be directly handled by the Mailbox server without hitting the CAS first.

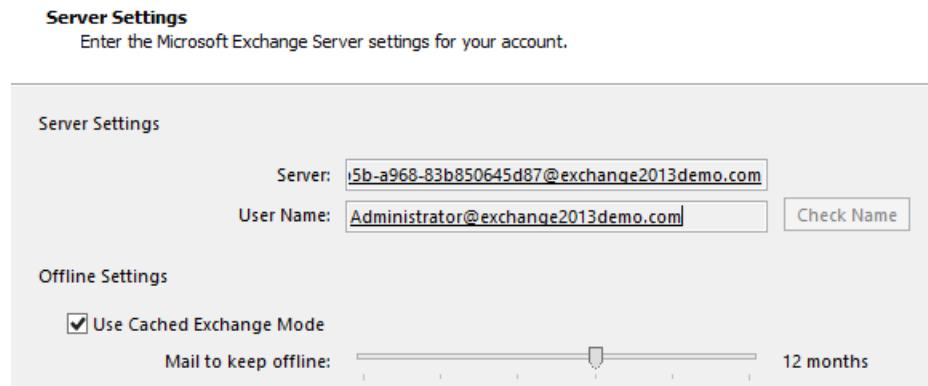
For more on UM refer to the Unified Messaging chapter of this guide.

## The Demise of RPC

With the RTM release of Exchange Server 2013, RPC-over-TCP connectivity – which has been the default connectivity model in previous releases of Exchange – was deprecated, leaving Outlook Anywhere (RPC-over-HTTP) as the only and default connectivity method for Outlook clients.

Removing the RPC-over-TCP layer also removed the need for a RPC Client Access Array namespace. As such, there's one less namespace to worry about and Microsoft was able to get rid of the pesky "Your administrator has made a change that requires you to restart Outlook" message whenever a database failover/switchover event occurs.

This change is also reflected in how Outlook reports which server it connects to. Instead of having a server name (RPCClientAccessServer), it will now show the GUID of the mailbox:



In Service Pack 1, Microsoft introduced a new connectivity option called MAPI/HTTP which seems to officially signal the end of RPC connectivity in future versions of Exchange Server.

MAPI/HTTP introduces a number of benefits over the traditional RPC/TCP and RPC/HTTP protocols. First, by removing the RPC layer in the communication between the client and the server, communications are simplified. This allows clients to (re)connect faster and even have more resilient connections; the RPC protocol wasn't really forgiving to network latency and outages.

Even though this change won't happen overnight, it's clear that HTTP-based connectivity is what Microsoft emphasizes for the future. At the moment MAPI/HTTP only works with Exchange 2013 SP1 in combination with Outlook 2013 SP1. Additionally it's disabled by default. Nonetheless it is going to become very important and you should definitely pay the necessary attention to it when sizing your Client Access servers, and when configuring load balancing.

## High Availability

The basic idea behind high availability for the CAS is to distribute the load amongst multiple servers. Your environment's resiliency comes from having more than a single server and the ability to automatically direct client traffic away from a failed server to a working one.

However, you will notice that configuring high availability for the Client Access server role doesn't involve much configuration of Exchange itself. The load balancing solutions on the other hand will be an important determining factor in how well your high availability solution will work and how it will be perceived by your users.

## Load Balancing Concepts

To begin with it's important to understand the basics of load balancing. Let's take a look at some of the concepts and terminology that you'll encounter when you're dealing with load balancing.

## Persistence (And Why it isn't Important in Exchange Server 2013)

In the world of load balancing, persistence – sometimes also called affinity - refers to maintaining a 1:1 relationship between the client and the server which is handling the client's request.

In previous versions of Exchange, some workloads (e.g. Outlook Web App or Exchange Web Services) required each subsequent client request to be handled by the same Client Access server that received the initial request. If not, things could break and the end user would get unexpected results.

Moving to a new Client Access server architecture allowed Exchange Server 2013 to eliminate the need for persistence for any of the workloads. This greatly simplifies the design of a highly available CAS infrastructure as well as the configuration of your load balancing solution.

Consequently, a challenge imposes itself - how does Exchange deal with authentication if an end user's connection can switch between different Client Access servers during a session?

When an initial client request reaches the CAS infrastructure, the CAS handling the request will authenticate the request and craft an authentication cookie for the client, which is then sent by the client with each subsequent request.

That authentication cookie is encrypted using the server's SSL certificate that is configured for IIS. If persistence were to be configured, those requests would reach the same CAS over and over. As a result, that CAS would be able to decrypt and read the authentication cookie, because it was encrypted with its own certificate in the first place.

However, as we mentioned earlier, Exchange Server 2013 doesn't require any persistence. This could cause subsequent requests to reach a different CAS. Without additional configuration, this would mean that subsequent requests potentially would need to be re-authenticated and as such result in an authentication pop-up or additional delay for handling the authentication.

The solution here is actually as simple as it is elegant. By configuring the same SSL certificate on each of the Client Access servers, you ensure that every CAS is able to decrypt the authentication cookie and thus can authenticate traffic without having to challenge the client for credentials.

**Note:** Regardless of how the Client Access server handles authentication, it's a best practice anyway to use the same SSL certificate on all Client Access servers.

Depending on the make and model of your load balancer you might find that the documentation describes persistent, and different ways to define the persistence settings when configuring the load balancer for Exchange Server 2013 traffic such as Source IP, Session ID or Cookie-based persistence.

It is fine to be aware of these settings and even use them if you wish to follow your load balancing vendor's guidance to the letter. But given the lack of need for these options in Exchange 2013, we won't discuss them any further here.

## Load Balancing Mechanisms

Load balancing mechanisms – sometimes referred to as scheduling methods - define how a load balancer decides which server in a group will receive incoming traffic. Based on the load balancer model you are using, you might have different options available to you. Some of the commonly available mechanisms are:

- **Round Robin** - The load balancer will cycle through all servers in the array and forward traffic to each CAS, as it reaches the load balancer. Once the load balancer has cycled through the list of servers, it will start with the first server again.
- **Least Connections** - The load balancer maintains a table in which it keeps track of how many connections were sent to each server. The server that has received the least connections so far, will receive the new connection.
- **Weighted Connections** - This is basically the same principle as Least Connections. Only with this method, you can assign a different priority ("weight") to each server in the array. Servers with a higher priority will then receive a higher percentage of connections.

**Real World:** Some load balancer vendors offer variations to the above scheduling methods. Some are quite complex but may also be a better match for your environment. This is especially true if not all servers in your environment are equally equipped – even though the recommendation and best practice is to use the same servers. Exchange itself doesn't care what scheduling method you use. It's up to you to decide which method fits your needs best.

**Note:** If you are not using a load balancer (virtual or hardware) and opt for Windows Network Load Balancing (WNLB), you won't be able to configure the load balancing mechanism. WNLB uses its own algorithm to define which server will receive the traffic. This algorithm is based on a fixed percentage distribution between the hosts in the cluster. This being said, we strongly recommend against using Windows Network Load Balancing for a number of reasons; one of them being the inability to use WNLB with multi-role servers participating in a Database Availability Group.

## Health Checking

One of the key benefits of load balancing is the ability to automatically detect when a server is unavailable or maybe not functioning correctly and then ignore that server for future connections until it becomes available/healthy again. The challenge exist in accurately determining when a server is healthy, and when is it not healthy.

Depending on the load balancing solution that you choose you might have different health checking options available to you.

One of the most limited solutions is Windows Network Load Balancing. WNLB will only check the host's availability by verifying if the hosts in the cluster are still responding to incoming connections. While this is probably better than having no health check at all, it doesn't reveal any information about the application which is running on those hosts (Exchange Server 2013). As such, this type of health checking is sometimes also referred to as being *application unaware*.

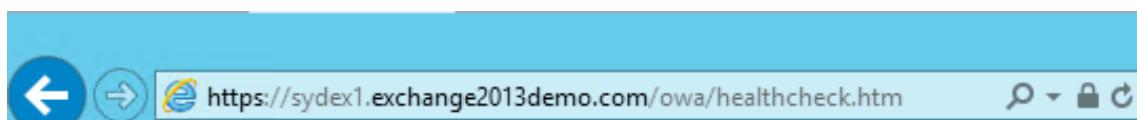
Of course, it is better to perform *application aware* health checks. The challenge here is that this approach typically isn't easy. Over time, load balancer vendors have offered many different options to achieve this goal.

The more advanced solutions will allow you to setup complex scripts which, for example, will execute a synthetic transaction and verify whether it completed successfully or not. If it did, the server is deemed healthy. If not, it's considered unhealthy. While this approach definitely is better than just checking a host's availability, it doesn't necessarily reflect the true state of the application or of all the workloads.

In order to solve this puzzle Microsoft worked together with different vendors and has created a solution in Exchange Server 2013 which tightly integrates with Exchange's built-in 'monitoring' system called Managed Availability.

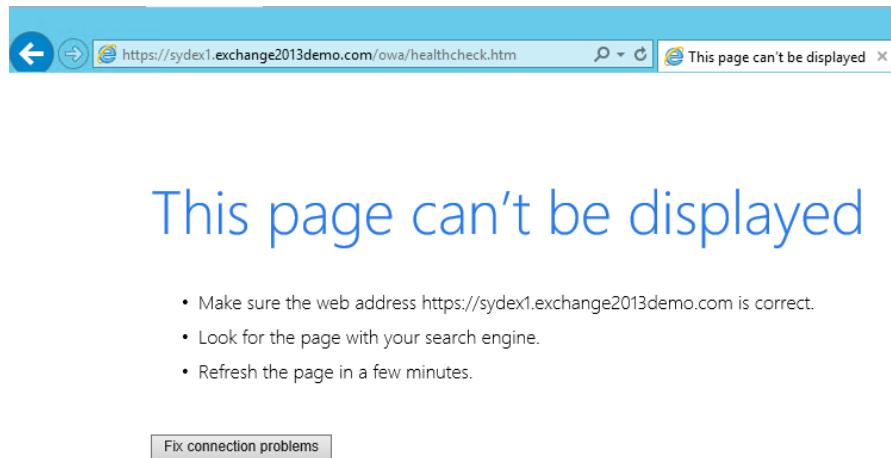
Thanks to this Managed Availability feature, every workload now has its own health check page. If a Managed Availability health probe determines that a workload is healthy, a health check page for that workload will be dynamically generated.

For example, the health check page for Outlook Web App can be seen at  
<https://<servername>/owa/healthcheck.htm>.



When the workload is unhealthy the page is not generated. As a result, the load balancer will not be able to open the page and will receive a 404 error instead.

The load balancer interprets this as an unhealthy server and stops sending client traffic to that server.



It's important to understand that the healthcheck.htm page – as it's being generated dynamically – isn't a file you will find on the file system. If you navigate in Windows Explorer to the root of the virtual directory, you will not find it; not even when the virtual directory is deemed healthy.

C:\Program Files\Microsoft\Exchange Server\V15\ClientAccess\Owa			
Name	Date modified	Type	Size
1x1	20/02/2014 17:09	GIF image	1 KB
anonymousecalendar.aspx	20/02/2014 17:06	ASPX File	4 KB
casredirect.aspx	20/02/2014 17:06	ASPX File	5 KB
csdc	20/02/2014 17:09	GIF image	1 KB
default.aspx	20/02/2014 17:06	ASPX File	29 KB
disabled.aspx	20/02/2014 17:06	ASPX File	1 KB
global.asax	20/02/2014 17:06	ASAX File	1 KB
info.aspx	20/02/2014 17:06	ASPX File	4 KB
InitSuiteServiceProxy.aspx	20/02/2014 17:06	ASPX File	1 KB

By configuring your load balancer to check whether or not the healthcheck.htm page is available, you effectively create a very granular and application-aware health check that leverages the intelligence of Managed Availability.

You can learn more about Managed Availability in the Managing and Monitoring chapter of this guide.

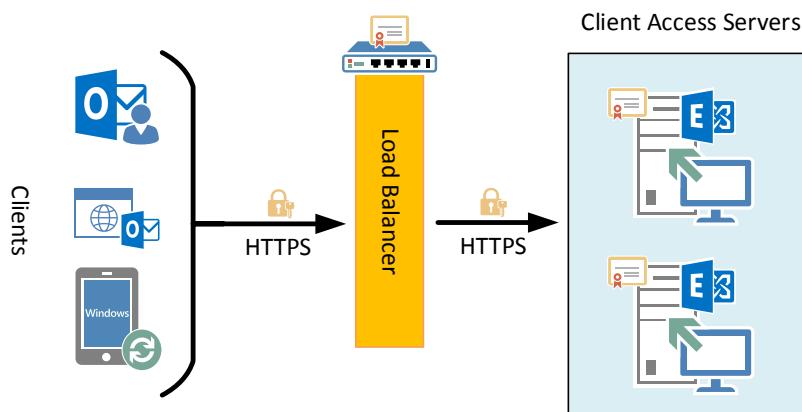
Now, having this health check page doesn't necessarily solve all issues. In the above example we accessed the health check page for OWA. This page only reflects a server's OWA health. It does not guarantee that any of the other workloads are working correctly.

If you don't want to use a single workload's health to determine whether a server should receive connections, you will have to configure multiple health checks per server; one per workload more specifically. This approach will be described later in this chapter when we talk about the different CAS HA models.

## SSL Bridging and SSL Offloading

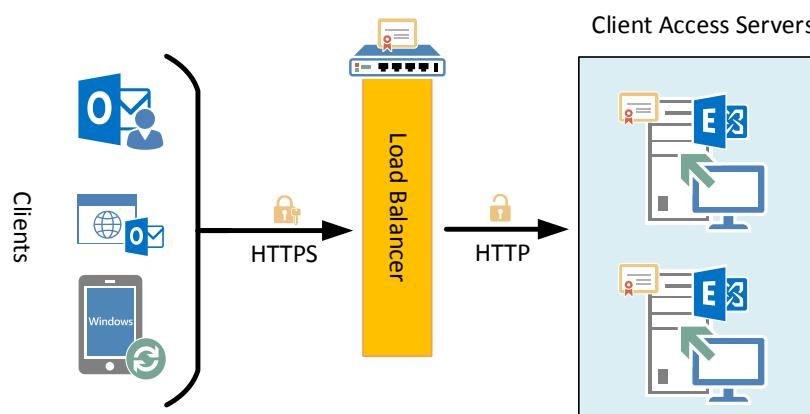
*SSL bridging* is when a load balancer decrypts traffic (e.g. to perform content inspection or make specific routing decisions) but then re-encrypts traffic when it forwards connections to the servers in the array. The Exchange servers still receive inbound HTTPS traffic and will have to decrypt it before they can process the data inside the stream.

SSL bridging puts a higher load on the load balancing solution as it will have to both decrypt/re-encrypt traffic. Additionally, you have to configure an SSL certificate on the load balancing solution which matches the Exchange server's published URLs.



**Note:** Requiring SSL is the default behaviour for most Client Access server workloads. Since the removal of RPC-over-TCP, Outlook Anywhere now also has an internal URL that by default also accepts unencrypted (HTTP) traffic, which is still secure because the traffic is protected by RPC encryption.

*SSL offloading* is a slightly different approach. SSL decryption is offloaded from Exchange and handled solely by the load balancer. When traffic is forwarded to the Exchange servers, it is not re-encrypted. The Exchange servers no longer have to decrypt traffic.



Although this seems like a big win (you save some CPU cycles which are otherwise spent to decrypt incoming traffic), there are some caveats.

First of all, SSL offloading wasn't supported until Exchange Server 2013 Service Pack 1. This means that if you're not running Service Pack 1 yet, you can't configure SSL offloading.

Secondly, not all workloads support SSL offloading. More specifically the MRS Proxy service which is responsible for handling cross-forest mailbox moves (like e.g. moves from/to Exchange Online) expects traffic to be encrypted. There is no way of changing this behavior in Exchange and thus the exception has to be configured at the load balancing solution level.

Although SSL offloading is a valid option, it is not recommended because it requires you to modify Exchange Server 2013's default behavior. Making such changes means that you are introducing complexity, thus increasing the efforts to setup, maintain and support the solution. Even though configuring SSL offloading in Exchange Server 2013 is a tad simpler than in Exchange 2010 it's still something that has to be done manually, which increases the risk of human error.

**Note:** Because the recommendation is not to use SSL offloading and because the use cases are rather specific, we will not discuss how to configure it here. Additional information on how to configure SSL Offloading for Exchange 2013 can be found on the following page: [http://technet.microsoft.com/en-us/library/dn635115\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/dn635115(v=exchg.150).aspx)

## Layer 4 / Layer 7 Load Balancing

When you read through load balancing documentation, you will see references to Layer 4 and Layer 7 load balancing. These terms refer to layers in the OSI model and describe how a load balancer responds to client requests.

#	Layer Name	Protocol(s)	Examples
7	Application	HTTP, FTP, SMTP	
6	Presentation	JPEG, GIF, PNG	
5	Session	AppleTalk, WinSock	
4	Transport	TCP, UDP	
3	Network	IP, ICMP, IPX	Router
2	Data Link	Ethernet	Switch
1	Physical	Ethernet, Token Ring	Repeater

The Transport layer is primarily responsible for making sure that the packets are sent between nodes. If a load balancer is configured to operate at Layer 4, this means that it will make its routing decisions purely based on the properties of the TCP connection; the IP Address and port number.

A virtual service (which represents the virtual entry point for the load balanced application), always consists of a unique combination of an IP address and a TCP port or port range. For example, 192.168.10.10:443.

When operating at Layer 4, this is all the load balancer cares about. Traffic that hits this specific virtual service will always be forwarded to the servers in the array regardless of the URL that you used or even which virtual directory you are connecting to. Even if the connection has nothing to do with Exchange, the load balancing solution will still forward the traffic to the back-end servers which will then handle the erroneous request by issuing an error.

If a load balancer is configured to operate at Layer 7 – the Application Layer – things are literally taken a few levels higher. When operating at this level, a load balancer will make what I like to call “informed decisions” about whether or not it should forward connections and to what servers it should forward those connections to.

The load balancer cannot make these informed decisions without inspecting the traffic. As traffic in Exchange 2013 is encrypted by default (HTTPs) the load balancer must perform SSL decryption before it can operate at this layer.

## SSL Certificate Requirements

HTTPS is the main protocol used for client-to-server and server-to-server communications in Exchange 2013. Because of that, the proper configuration of certificates is extremely important. One of the aspects of configuring certificates is to ensure that the certificate you will use covers the namespaces you use in your deployment.

A namespace, in the context of Exchange 2013, usually refers to the fully qualified domain name(s) (FQDN) clients use to connect to Exchange. The namespaces for your Exchange server are the FQDNs that are in the URLs configured on virtual directories, such as the OWA virtual directory. For example, an OWA URL of <https://mail.exchange2013demo.com> means that “mail.exchange2013demo.com” is one of your Client Access server namespaces.

## Namespace Planning

The general rule for namespace planning is to keep things as simple as possible, and the new Exchange Server 2013 architecture makes this possible.

By having removed the RPC-over-TCP feature from Exchange 2013, Microsoft removed the need to plan for an RPC CAS Array namespace. The new CAS architecture also removes the need to have an OWA fallback URL in a site-resilient deployment.

These and other changes mean that a highly available, site-resilient Exchange 2013 deployment only requires a minimum of two distinct namespaces:

- Autodiscover
- Internet Protocol Namespace

There are different ways in which you can deal with namespaces. Largely influenced by your physical topology, you can choose from either the *bound model* or the *unbound model*.

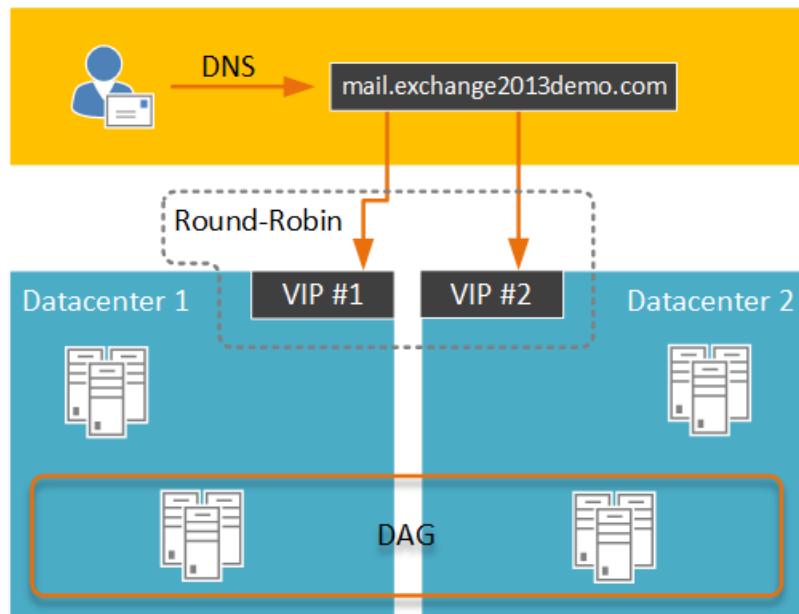
#### *Unbound Model*

This is the scenario in which a single Database Availability Group is stretched across multiple datacenters and a pool of Client Access servers (in either datacenter) processes client traffic for both datacenters.

This model is called the *unbound model* because none of the datacenter's connectivity (pool of Client Access servers) is limited (bound) to the site in which they are deployed.

With the unbound model clients can connect to a Client Access server in either datacenter regardless of where their mailbox is hosted in the DAG at any given time. Traffic will automatically find its way to the destination Mailbox server hosting the active copy of the database that has the user's mailbox, even if that mailbox resides in the other datacenter.

The unbound model means that you only require a single namespace for both datacenters.



Based on the unbound model, for an organization using the domain name “exchange2013demo.com” the certificate you would purchase would contain at least the following namespaces:

- exchange2013demo.com
- autodiscover.exchange2013demo.com
- mail.exchange2013demo.com

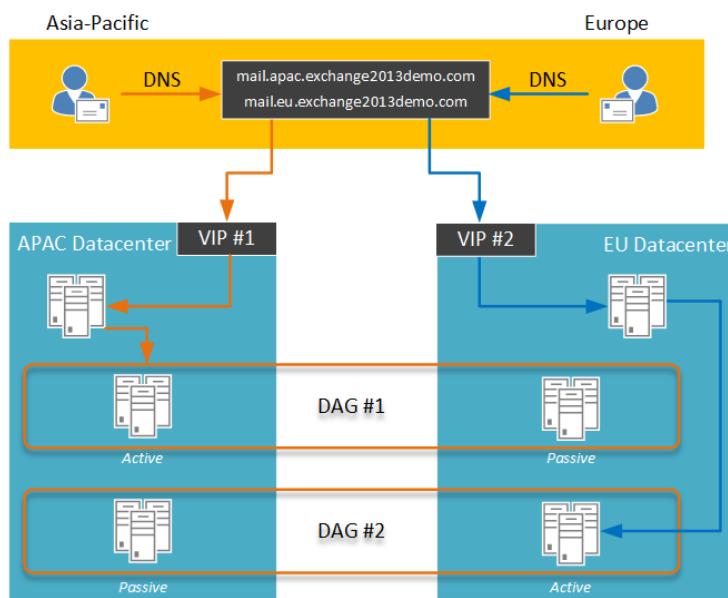
**Note:** This example assumes that “mail.exchange2013demo.com” is the desired alias for end users to use when connecting to services such as Outlook Web App and ActiveSync, and is configured as the URL on the Client Access server virtual directories. You can of course choose any alias that you like, such as “webmail” or “remote”.

The unbound model is simple to deploy and manage, however there are situations in which you might want to limit the amount of traffic flowing between datacenters, especially if the network between these datacenters is costly. This could be the case when you have datacenters in different geographic regions around the world. In such case it might make more sense to deploy a bound model.

#### *Bound Model*

In the bound model you associate a specific set of users to a specific datacenter. To achieve this, you would typically deploy a Database Availability Group, stretched across two datacenters, but have it host the active mailbox databases out of a single datacenter – at least during normal operations.

This is also referred to as an active/passive DAG model. If you are looking to host active mailboxes in both datacenters, you could deploy a second Database Availability Group where the active and passive roles are reversed. Each Database Availability Group then preferably hosts mailbox databases in a different datacenter:



The bound model requires multiple namespaces, preferably two per datacenter; one for the primary namespace and one for the fallback namespace.

The fallback namespace, just as the name suggests, is only used when switching operations back to the primary datacenter after a previous datacenter switchover. As part of the switch back, DNS records have to be modified to point once again to the primary datacenter.

During this change – and until the DNS records TTL expired – clients might still connect to the secondary datacenter, because of the cached DNS record on the computer that still points to the second datacenter. When this happens, the client will be redirected to the primary datacenter. However, because you have performed the switch back, this would point to the exact same namespace, effectively redirecting the client back to the secondary datacenter.

If no mechanism would exist, this might potentially create a redirection loop. In order to mitigate this particular behavior, Exchange uses the fallback namespace. When it detects that a client tried to login in the secondary datacenter and that it's being redirected back to the secondary datacenter, it will then switch to redirecting the client to the fallback namespace instead.

As part of the DNS guidance, Microsoft recommends setting a TTL of 5 minutes for all Exchange-related DNS records. Theoretically, this means that during a switch back to the primary datacenter this condition should only be possible between the DNS changes and the expiry time of the DNS records – being five minutes if you followed the guidance.

Unfortunately, browser clients do not always honor the TTL of a DNS record. They often use a client-cache which can last much longer and thus potentially extend the time frame within which this scenario could happen. Hence why the fallback namespace is very useful.

Based on the bound model, the certificate you would purchase for the example above would contain at least the following namespaces:

- exchange2013demo.com
- autodiscover.exchange2013demo.com
- mail.eu.exchange2013demo.com
- mail-fallback.eu.exchange2013demo.com
- mail.apac.exchange2013demo.com
- mail-fallback.apac.exchange2013demo.com

## Split-DNS

Ever since Exchange 2007, Microsoft has recommended to deploy a split-DNS infrastructure whenever possible. This means that you use the same namespace internally and externally, but have different IP addresses for that namespace depending on where your client is located; inside or outside the corporate network.

There is no change in this for Exchange 2013. It is still recommended to deploy a split-DNS infrastructure as it greatly simplifies Exchange's configuration and reduces the number of namespaces you need on the certificate.

If you decided not to use split-DNS or you are unable to do so, Exchange 2013 now has the ability to configure a different internal hostname for Outlook Anywhere. In prior releases, Outlook Anywhere only had an external namespace which made it extra difficult in scenarios where you also wanted to use Outlook Anywhere internally, but couldn't deploy split-DNS.

In a split-DNS scenario the internal and external namespace exist in the same DNS domain, and are usually configured to the same URL, for example "mail.exchange2013demo.com".

Even though this is the recommendation configuration it creates additional challenges with regards to how clients authenticate to Exchange for Outlook Anywhere.

Outlook Anywhere can be configured to use different authentication methods for internal and external FQDN. But if both namespaces are identical Exchange has no way of knowing whether a request originated from within the network or not. This because the Outlook client – once it has received the Autodiscover response - will always first try connecting to the Internal hostname for Outlook Anywhere and succeed in a split DNS environment.

As such, it will always assume that the connection is internal, even if it is actually originating from outside the network, and will only use the authentication method configured for the internal Outlook Anywhere namespace.

If your organization needs different authentication methods for internal and external Outlook Anywhere connections then you will need to configure different Outlook Anywhere hostnames, for example "mail.exchange2013demo.com" and "mail-internal.exchange2013demo.com". You can then use the split DNS zones to ensure that only the internal hostname is resolvable internally, and only the external hostname resolvable externally. You will also need to ensure that both Outlook Anywhere names are included on the SSL certificate.

**Real World:** Sometimes organizations cannot easily deploy split-DNS. After all, changing over to split-DNS can be both time consuming and a little risky at the same time. A possible workaround is to use PinPoint DNS zones. PinPoint DNS zones are zones in DNS that point to a single hostname instead of having a zone for the domain and adding a DNS record per host. When split-DNS is not possible, PinPoint DNS zones are a simple alternative which can be implemented quickly with little to no risk.

# POP, IMAP & SMTP

For the POP, IMAP and SMTP protocols it's generally recommended to have a separate namespace per protocol. For example:

- pop.exchange2013demo.com
- imap.exchange2013demo.com
- smtp.exchange2013demo.com

Although the use of POP and IMAP is declining, they are still important client connectivity methods which shouldn't be forgotten. However, there is no requirement to include the namespaces for these two workloads on the certificate which you use for Exchange – that is unless you choose to use Secure POP or Secure IMAP.

**Real World:** Given that POP and IMAP are losing popularity, many organizations that still require to support these protocols for legacy applications choose not to implement a different namespace. Instead they use the same namespace as the HTTP services such as OWA and Outlook Anywhere.

If you want to be able to force TLS or you have a hybrid deployment with Office 365, you must ensure that the SMTP namespace is included on the certificate. Although it is recommended that this is a separate namespace to other protocols such as HTTP and POP, there is no requirement to use a different namespace. You could just as easily choose to use the HTTP namespace for SMTP too.

If you choose to use a different namespace, you can either add it to the certificate which you also use for the HTTP workloads or you can purchase a separate certificate for SMTP.

**Real World:** When you deploy a Hybrid Exchange configuration, you are required to use a third party certificate for both IIS and SMTP. Sometimes the SMTP certificate you configure might not be recognized by the Hybrid Configuration Wizard. Mostly, this is because the wizard expects certain attributes of the certificate to have a very specific value.

For instance, the **RootCAType** attribute should always be **ThirdParty**. This is because a hybrid configuration requires you to use certificates from a trusted, Public Certificate Authority. If you want to find out what the value for a certificate is, use the Get-ExchangeCertificate cmdlet.

# Deciding on a Load Balancing Model

Part of designing a highly available Exchange 2013 environment is to decide which load balancing model you will be using. This decision can be influenced by a variety of factors, including the fact that you might or might not already have some load balancing solutions which you can re-use for Exchange.

Next to some of the technical elements which can influence your decision, there's also the business and SLA requirements to which your solution has to adhere.

Below you will find an overview of the different load balancing models, outlining both the pros and cons of each.

## DNS Round Robin

DNS round robin is the least costly load balancing solution and requires the least effort to setup. In fact, you rely entirely on DNS and how client computers and devices deal with multiple DNS records for the same host.

The principle is simple; you associate each Client Access server with the Exchange namespace. You do this by adding an A-record to your DNS zone for each of the Client Access servers.

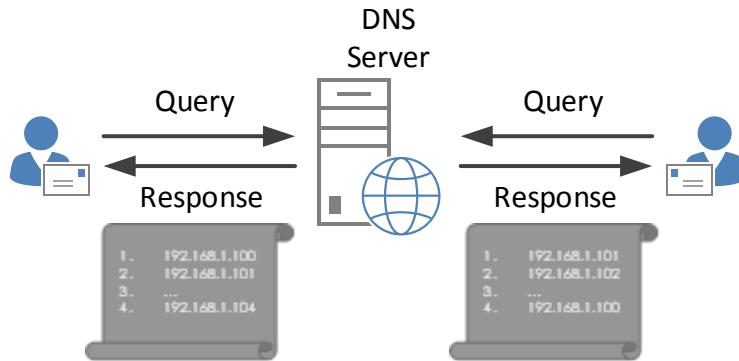
If there are two Client Access servers then you add two A records, one for each Client Access server IP address. If there are three Client Access servers, you add three A records. And so on.

In this example there are four servers deployed, and a DNS record for the HTTP namespace has been added for each server's IP address.

Name	Type	TTL	Section	IPAddress
---	---	---	-----	-----
mail.exchange2013demo.com	A	3600	Answer	192.168.1.101
mail.exchange2013demo.com	A	3600	Answer	192.168.0.102
mail.exchange2013demo.com	A	3600	Answer	192.168.0.101
mail.exchange2013demo.com	A	3600	Answer	192.168.1.102

The result is that whenever a client performs a DNS query for the Exchange namespace, it will receive an ordered list of all the different A records for that specific namespace.

With each subsequent DNS query, that list will get reordered so that the next Client Access server in the array is now the first entry in the DNS response:



The biggest benefits of DNS round robin is that it requires very little effort to setup and that there is no requirement for an external virtual or physical load balancing solution.

However, DNS round robin has some important drawbacks that you should be aware of. One of them being that it does not perform any health checks.

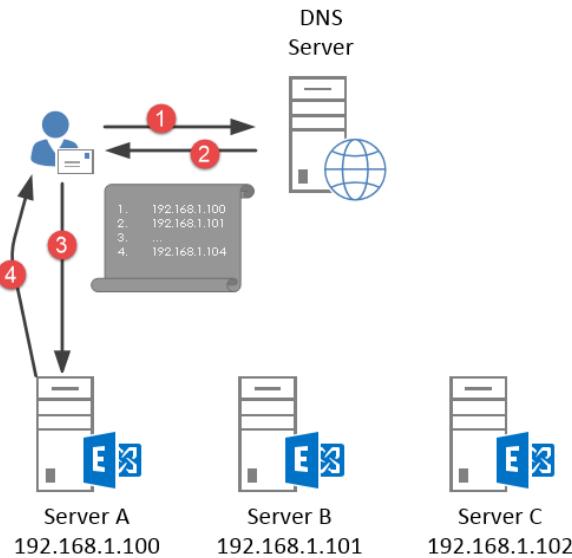
Even if a server is down for maintenance, the DNS server has no way of knowing that it's unable to service client requests and it will still hand out that server's IP address to client DNS requests.

As long as the server's IP address is not the first on the ordered list the clients receives from the DNS server, there will not be a problem. But as we described before, because of the way DNS Round Robin works, clients will sooner or later get a DNS response in which this particular server's IP address is on top of the list.

To better understand why this can be a potential issue, let's have a look at how clients deal with this scenario.

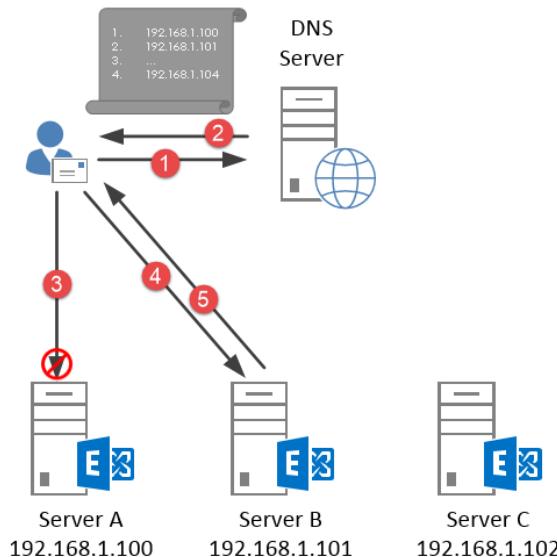
The following list depicts the interaction between a client, the DNS server and the Exchange Client Access servers:

1. Client performs DNS query for the Exchange namespace. For example:  
mail.exchange2013demo.com
2. DNS server responds with an ordered list. For example: Server A's IP address is on top of the list, Server B's second, Server C's third etc...
3. The client connects to the **first** IP address from the list (Server A)
4. The Client Access server (Server A) responds to the client's request.
5. The client will continue to "talk" to Server A for as long as it's available or until the DNS record's Time-To-Live (TTL) expires. The latter will force the client to perform a new DNS query and possibly get a differently ordered list.



So far, so good. Let's have a look at what happens if Server A is completely unavailable (e.g. offline).

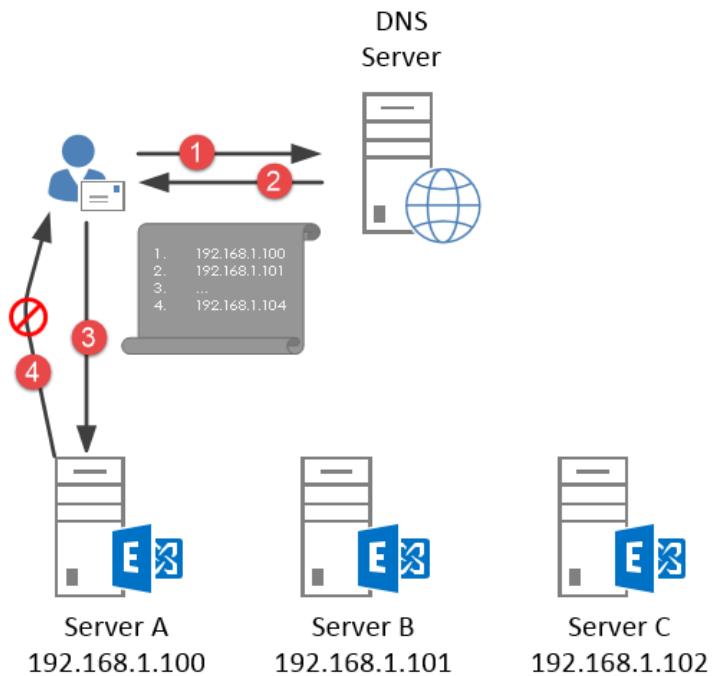
1. Client performs DNS query for the Exchange namespace
2. DNS server responds with an ordered list. For example: Server A's IP address is on top of the list, Server B's second, Server C's third etc...
3. The client connects to the **first** IP address from the list (Server A)
4. The server is offline and thus the client cannot connect to the server
5. The client will now cycle to the second entry on the list and try to connect to that server (Server B).
6. The server (Server B) responds to the client's request.
7. The client will continue to "talk" to Server B for as long as it's available or until the DNS record expires. The latter will force the client to perform a new DNS query and possibly get a differently ordered list.



Again, no issue there.

Now, let's have a look at what happens if Server A is still connected to the network (and available) but Exchange is misbehaving for some reason:

1. Client performs DNS query for the Exchange namespace
2. DNS server responds with an ordered list. For example: Server A's IP address is on top of the list, Server B's second, Server C's third etc...
3. The client connects to the **first** IP address from the list (Server A)
4. The Client Access server (Server A) responds to the client's **connection request**. However, Exchange is currently broken and does not respond correctly to the client's request.
5. The client will not cycle to the next server in the list. Instead it will have a broken experience.



As you can see, this isn't really the behavior we are looking for. The reason that this happens isn't because of Exchange, but rather in how clients in general deal with DNS Round Robin. As long as a client can make a connection to the server, it will not cycle to the next server in the list. It will only try a different IP address if the client encounters a hard TCP connect failure.

Because DNS Round Robin does not do any health checking of the server, there is no reliable way of telling what server should or should not receive connection. When a server is entirely unavailable and clients are not able to connect to it, DNS Round Robin will likely work relatively well.

This could for example also be the case when the IIS server is stopped on the Exchange Server as clients won't be able to make an HTTP(s) connection and thus automatically cycle to the next server. In a scenario where IIS is still running but there's some other underlying issue, like one of the workloads not working as expected, clients will still connect to that failed server but have a broken experience.

In a scenario where you find that a server has failed, you would remove the server's IP address entry from DNS effectively stopping it from being handed out to clients and thus preventing client from connecting to it.

Unfortunately this is a manual interaction. Also, there might be several hours in between the time that you determine there is an issue and you successfully remove the entry from DNS.

As a rule of thumb, if you choose to use DNS Round Robin, set the DNS record's Time-To-Live (TTL) option to a value as low as 5 to 10 minutes. This will ensure that if you ever have to add/remove a server from DNS, your clients will pick up the changes more quickly.

**Real world:** it's unlikely that you will come across a DNS Round-Robin only deployment. Instead, you will find that many organizations deploy a combination of DNS Round-Robin and load balancers.

Take the example of the unbound namespace, earlier in this chapter. Using DNS Round-Robin, you can distribute traffic across both of the datacenters. Each datacenter would then have a pair of highly available load balancers deployed which will make sure that no connections are forwarded to a failed Exchange server.

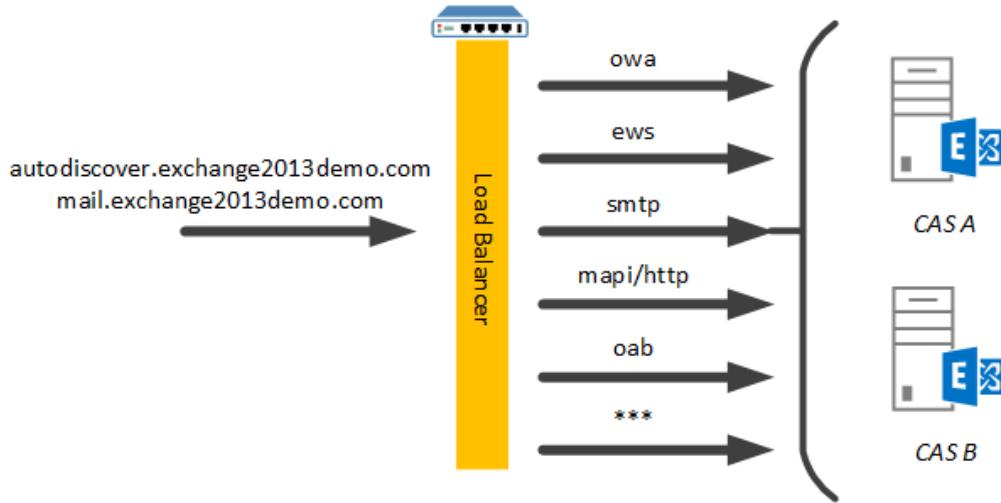
You would even be able to configure your load balancers to forward traffic to the other datacenter if all local Exchange servers are unavailable. Only if there would be something wrong with the load balancers in one of the datacenters, you would have to go into DNS to make changes manually.

## Layer 4 Load Balancing with a Single Namespace

As described earlier in this chapter, load balancing at Layer 4 means that we care only about the IP address and TCP port number. Because in this scenario we use a single namespace, we can also only have a single virtual service. Consequently, there are some limitations which we need to keep in mind:

1. The load balancer is unaware of the type of application traffic or the destination URL
2. We can configure only a single health check as there is only a single virtual service

The following image depicts the logical structure of such a setup.



The load balancer cannot make a distinction between the URLs a client is sending requests to. That would require use to decrypt traffic and thus move us to Layer 7 which is covered later in this chapter.

This means that all Exchange traffic for a specific protocol (e.g. HTTP) will hit the same virtual service and be subject to the same set of rules.

Another consequence of this, is that the health check becomes very critical. As we can only configure a single health check it is important to configure it to verify the health of the workload which is used the most in your environment.

After all, there is no purpose to configure the health check to verify the health of the OWA virtual directory if no one uses OWA, but everyone connects using Outlook Anywhere.

Consider the following scenario; a load balancer is configured to operate at Layer 4. It has a virtual service for all HTTP traffic and the health check is configured to verify the health of the OWA virtual directory.

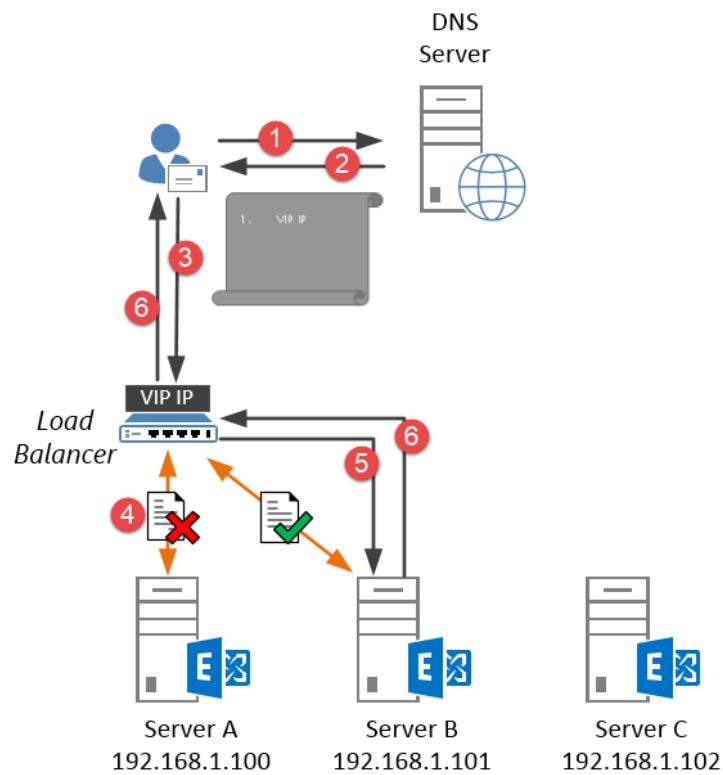
Let's see how clients interact with the server:

1. Client performs DNS query for the Exchange namespace
2. DNS server responds with the IP address of the virtual service (VIP)
3. The client connects to the IP address from the virtual service
4. The Load Balancer blindly forwards the traffic to one of the Client Access servers in the array. It counts on the server to decrypt the traffic and deal with the request.
5. The Exchange server which received the requests, responds to the client.

This scenario will repeat itself for each client trying to make a connection. Depending on the scheduling method of the load balancer, each incoming request will be forwarded to another Client Access server.

Now, let's see what happens if a server (e.g. Server A) has issues with OWA and a user tries connecting to OWA:

1. Client performs DNS query for the Exchange namespace
2. DNS server responds with the IP address of the virtual service (VIP)
3. The client connects to the IP address from the virtual service
4. Because the load balancer's health check has failed for Server 1, it will be taken out of service and will not receive client traffic.
5. The load balancer will blindly forward the traffic to one of the remaining servers in the array (e.g. Server 2). It counts on the server to decrypt the traffic and deal with the request.
6. The Exchange server which received the requests, responds to the client.

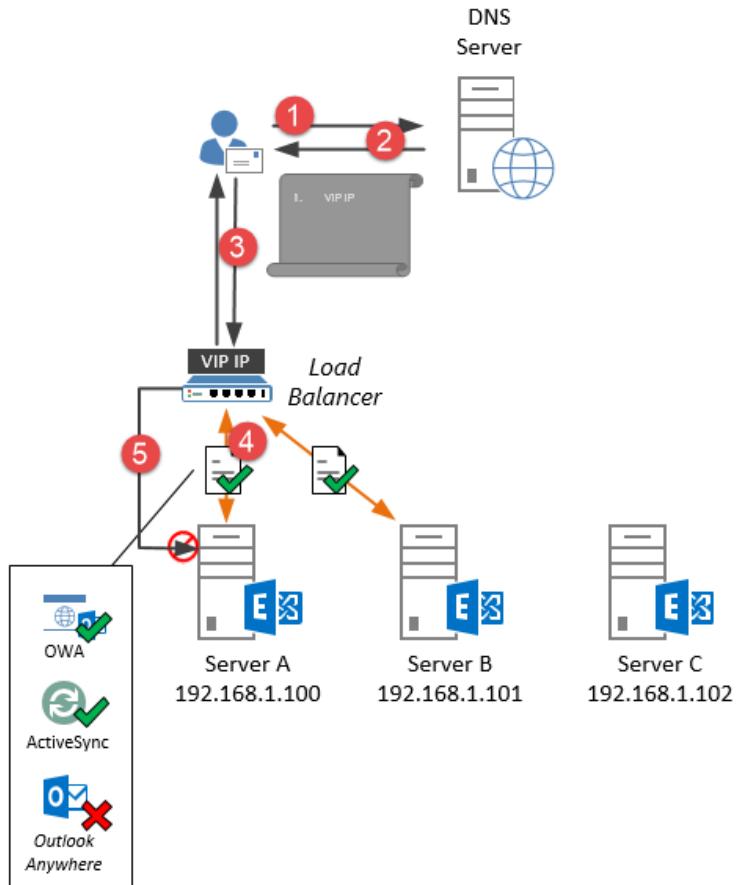


So far, so good. The health check did its job and took out of service a server which would have been unable to service the user's request.

In this third scenario, we will take a look at what happens if a server is having issues with Outlook Anywhere and the client is using Outlook to connect:

1. Client performs DNS query for the Exchange namespace
2. DNS server responds with the IP address of the virtual service (VIP)
3. The client connects to the IP address from the virtual service
4. The load balancer's health check will have completed successfully. Even though Server 1 has a faulty Outlook Anywhere component, OWA's health is still OK and as such the health check will deem the server as healthy as well.

- The load balancer will blindly forward the traffic to one of the Client Access servers in the array, possibly even the one with the failed Outlook Anywhere component. If the traffic gets forwarded to this server, the user's experience will likely be broken.



The single health check is probably the largest limitation in this scenario. While it will work fine most of the time, the outage of a single workload will immediately lead an entire server to be put out of service.

Also, there is no guarantee that a server will be taken out of service. For example, if you configure the health check to probe the OWA virtual directory, but there is an issue within the Outlook Anywhere feature (RPC virtual directory) you risk of having traffic being sent to that server regardless. Of course, this would then lead to a bad user experience if they are using Outlook Anywhere

**Real world:** Despite the additional value of having a health check per workload, many deployments are configured to use only a single health check. Many customers are willing to accept the risk in return for few namespaces, lower SSL certificate costs, and a simpler load balancer configuration.

## Layer 4 Load Balancing with Multiple Namespaces

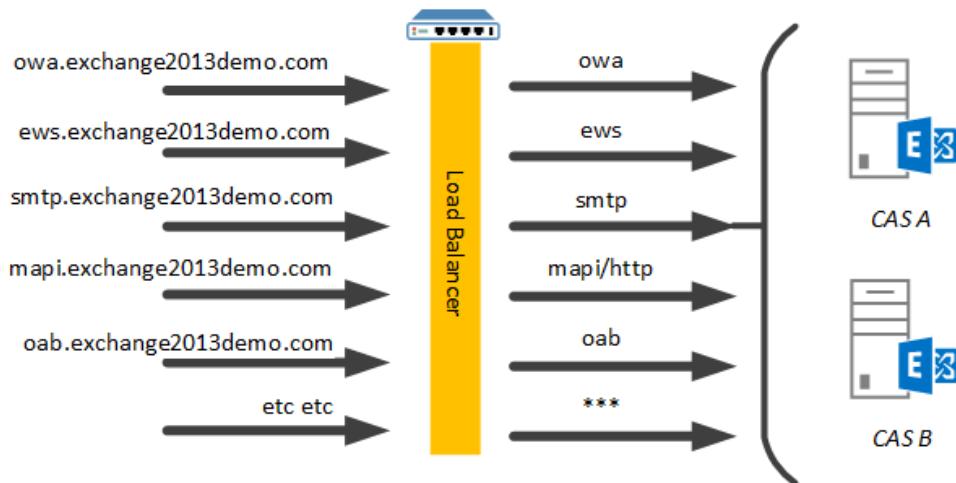
If you want to increase the resiliency of your solution by having a health check per workload, there are basically two options. One of them is to maintain the principle of balancing load at Layer 4, but to create a separate namespace per workload. This will allow you to define multiple virtual services; one for each workload and thus one health check per workload as well.

As with any solution, there are two sides to the story. While this solution will certainly increase your solution's resiliency by improving the effectiveness of the health checks, it will also increase the complexity and cost.

The reason being is that you will need a separate IP address per workload and multiple namespaces on the certificate, for example:

- owa.exchange2013demo.com
- outlook.exchange2013demo.com
- ews.exchange2013demo.com
- mapi.exchange2013demo.com
- oab.exchange2013demo.com
- eas.exchange2013demo.com
- ecp.exchange2013demo.com

The pool of publicly available IPv4 addresses is close to being depleted which makes an external IPv4 address a scarce resource. Even though you might already have a pool of IP addresses attributed to you, having to use a different IP address per workload may be an inefficient use of those resources. On a less technical note, having all these different namespaces doesn't look very good, does it?



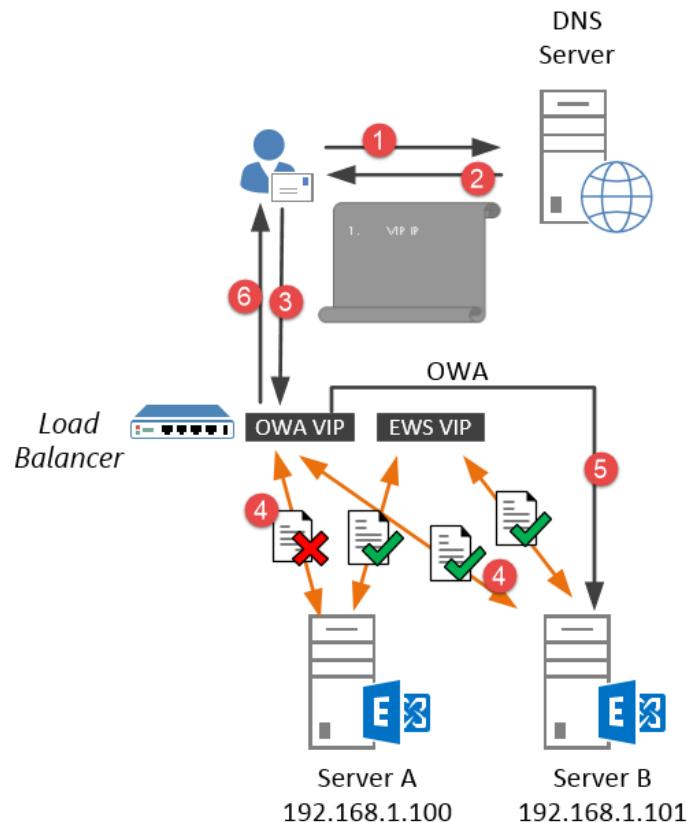
The interaction between a client and the load balancer and servers is similar to the previous scenario.

Assuming that all servers are working just fine, things would look like this:

1. Client performs DNS query for a specific Exchange workload's namespace. E.g. owa.exchange2013demo.com
2. DNS server responds with the IP address of the virtual service (VIP) on the load balancer
3. The client connects to the IP address from the virtual service
4. The load balancer's health check will have completed successfully.
5. The load balancer will forward the traffic to one of the healthy Client Access servers in the array.
6. The Exchange server which received the requests, responds to the client.

Now, let's have a look at what happens if one of the server's OWA components fail:

1. A client performs a DNS query for a specific Exchange workload's namespace. E.g. owa.exchange2013demo.com
2. The DNS server responds with the IP address of the virtual service (VIP) on the load balancer
3. The client connects to the IP address from the virtual service
4. The load balancer's health check will have completed successfully for all servers in the array except one. That one failed server is now taken out of service and will not be considered to send client OWA traffic to.
5. The load balancer will forward the traffic to one of the Client Access servers in the array that are still healthy.
6. The Exchange server which received the requests, responds to the client.



The above scenario would be the exact same for any and all of the workloads. Additionally, the decision to take a server offline is far more granular. Because there is a health check per workload, a server might be taken out of service for OWA, but it might still be used to service requests for Outlook Anywhere. A server will only be entirely unavailable if all health checks have failed or if the server is put into maintenance mode.

## Layer 7 Load Balancing

The second option to have multiple health checks per workload is to configure the load balancer to operate at Layer 7. By doing so, the load balancer can now access the HTTP conversation between the client and the server. Before it is able to do so, it needs to decrypt the incoming SSL traffic first.

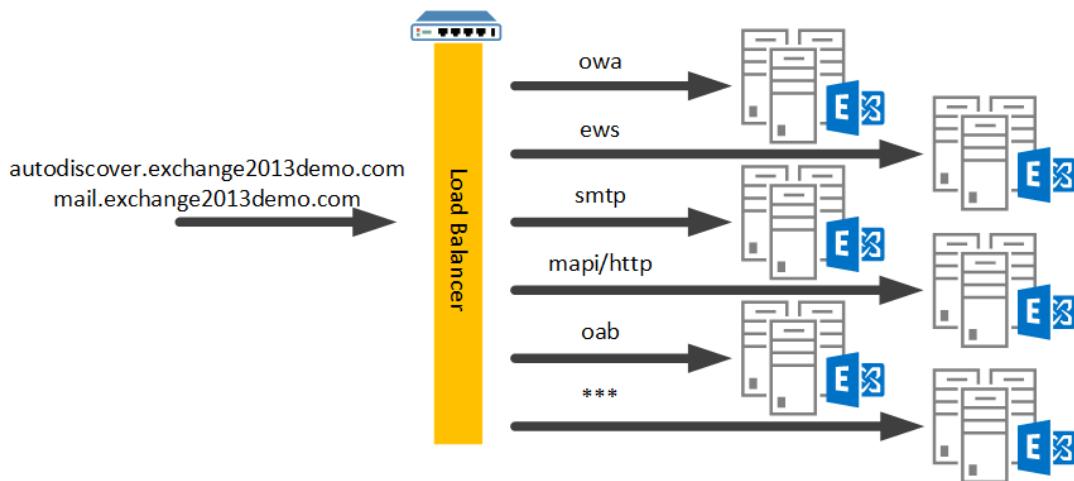
Once that is taken care of, the load balancer can perform various actions including:

- Dynamically routing traffic based on the destination URL
- Rewrite URLs as they pass through the load balancer

While the latter option can sometimes be handy in migration scenarios, it's mainly the first option that interests us at this time. Using a feature called content switching, which most load balancer vendors offer by default, you can configure a virtual service to send traffic to a specific set of servers based on the destination URL from the request.

For instance, a request for “<https://mail.exchange2013demo.com/owa>” could be sent to a different set of servers than a request for “<http://mail.exchange2013demo.com/rpc>” even though they're sharing a namespace and thus both use the same IP address.

**Real world:** The scenario described above is mostly theoretical. Typically, organization would use a single Client Access server pool for all workloads. The main reason to use content switching is the ability to more granularly control the health checks for each workload.



Depending on your load balancer, features might be named slightly different, but the main principle is the same. A single virtual service is configured with the IP address that corresponds to the namespace and a TCP port (443). Then, sub-virtual services are created for each of the different Exchange workloads:

- Outlook Web App (/OWA)
- Exchange Admin Center (/ECP)
- Offline Address Book (/OAB)
- Exchange Active Sync (/Microsoft-Server-ActiveSync)
- MAPI/HTTP (/MAPI)
- Outlook Anywhere (/RPC)
- Exchange Web Services (/EWS)
- AutoDiscover (/Autodiscover)

For each of the workloads a content rule (typically based on a regular expression) is created. This rule will later be used to analyze traffic and see if it matches with any of the aforementioned workloads.

The main virtual service will receive the client traffic, decrypt the traffic and then analyze it based on rules that have been created earlier. These rules will determine where traffic should be sent to.

The load balancer then forwards traffic to the appropriate sub-virtual service which contains all the specific configuration items such as the scheduling method.

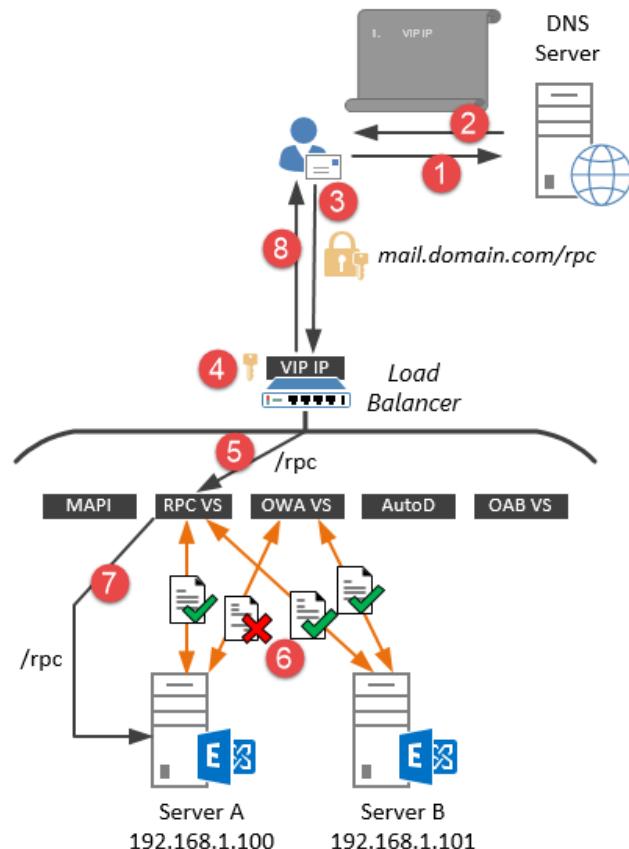
The series of steps below illustrate how a client interacts with a server in a layer 7 load balancing scenario where the client tries to access Outlook Web App (OWA):

1. The Client performs a DNS query for the Exchange namespace. E.g.  
mail.exchange2013demo.com
2. The DNS server responds with the IP address of the virtual service (VIP) on the load balancer
3. The client connects to the IP address from the virtual service
4. The load balancer's decrypts the traffic and analyzes the destination URL using the content rules that were created earlier
5. The load balancer will determine that the URL "https://mail.exchange2013demo.com/owa" belongs to the OWA sub-virtual service and forward traffic to that sub-virtual service
6. The OWA sub-virtual service is configured with a specific health check which monitors the healthcheck.htm page for OWA for each of the servers in the array. Given that all servers are healthy, the check completes successfully.
7. The load balancer now forwards the traffic to one of the servers in the array.
8. The Exchange server which received the requests, responds back to the client.

Consider the same example as above, only this time one of the server's OWA has failed and the client tries connecting using Outlook Anywhere.

For demonstration purposes let's assume that the health check for the RPC virtual directory completed successfully for all the servers in the array.

1. The Client performs a DNS query for a specific the Exchange namespace. E.g. mail.exchange2013demo.com
2. The DNS server responds with the IP address of the virtual service (VIP) on the load balancer
3. The client connects to the IP address from the virtual service
4. The load balancer's decrypts the traffic and analyzes the destination URL using the content rules that were created earlier
5. The load balancer will determine that the URL "https://mail.exchange2013demo.com/rpc" belongs to the RPC sub-virtual service and forward traffic to that particular sub-virtual service
6. While the OWA health check for one of the servers has failed, the RPC health check for that and the other servers will have completed successfully. Rather than taking that entire server out of service it will now still receive connections for all workloads except OWA.
7. The load balancer now forwards the traffic to one of the servers in the array; possibly the server for which the OWA health check failed.
8. The Exchange server which received the requests, responds back to the client.



Similar to the scenario of layer 4 load balancing with multiple namespaces, the failure of a single health check has no impact to the other workloads on a particular server.

The end result is the same, but we didn't use as many namespaces or IP addresses which makes this solutions simpler from an Exchange point-of-view. From the load balancer's standpoint however, this specific setup will be more complex.

Despite the additional work on the load balancer, the layer 7 option is the most flexible and takes advantage of Exchange Server 2013's new health check features without rendering the deployment of Exchange more complex.

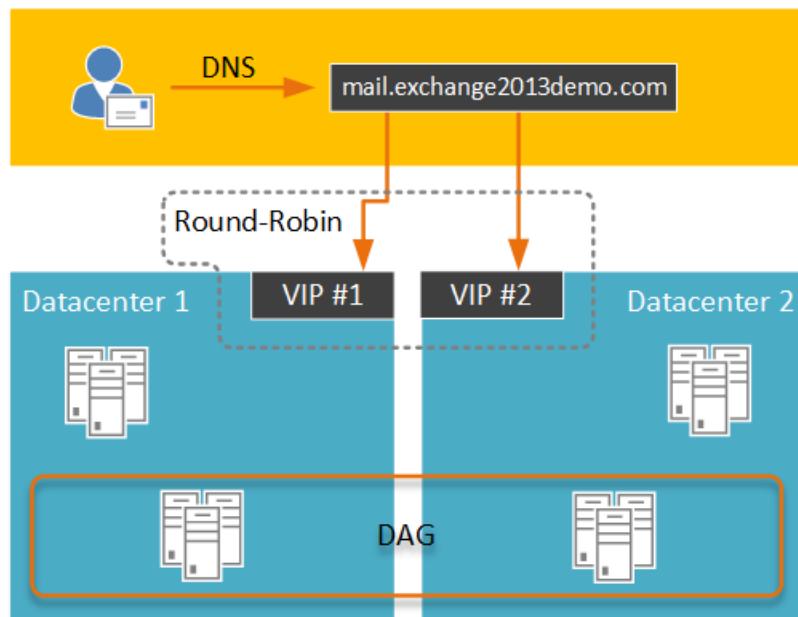
As such, I would always recommend choosing this scenario over any of the previous – that is, if your load balancer supports it and is able to handle the additional load.

## What About Multi-Site Load Balancing?

Multi-site deployments bring specific challenges with them and the solutions depend on the approach you have taken with namespace planning; unbound or bound.

### DNS round-robin

With the unbound model you are less concerned about which datacenter the client traffic gets sent to. As such, DNS round-robin is an ideal solution to divide traffic amongst two data centers; roughly half of your users will hit one datacenter, the other half will hit the second datacenter.



If the connectivity between both datacenters is both fast and cheap, this will not really be a concern. However, there are times in which this is not the case and you do care to what datacenter traffic is sent.

For instance, if one of the datacenters is solely used as a disaster recovery site or when you want to make a distinction of where traffic is sent based on the client's location; the latter being the case when you have datacenters in both Europe and Asia Pacific regions, and you want clients in Europe to hit the datacenters in Europe and clients in APAC to hit the APAC datacenters.

One way to deal with this requirement is to use the bound namespace model instead of the unbound model. Alternatively, you can use something called Geo DNS.

## Geo DNS

Geo DNS will hand out IP addresses to clients based on the client's geographic location. The best way to explain this is to use an example.

If a client is located in Europe and it performs a DNS query for the Exchange namespace, the Geo DNS service will reply back with the IP addresses of the datacenters, but it will make sure that the European datacenters are on top of the ordered list that the client receives.

Similarly, it will make sure that the Asia Pacific datacenter's IP addresses are on top of the list if a client from the APAC region performs a DNS query.

**Real World:** Geo DNS solutions use public and commercial lists that map IP addresses to a specific geographic location. Often these lists are very specific, potentially even down to the city level.

This solution will not prevent a user from Europe who is traveling in APAC from connecting to an APAC datacenter. If you really want to segregate traffic based on the user's default region, you should deploy a bound namespace.

**Note:** By using the bound namespace model, you lose the ability to automatically switch-over from one datacenter to the other.

In the unbound model, a connectivity issue or potentially the loss of a datacenter will automatically cause clients to connect to the VIP of the second datacenter, provided a possible client-side timeout of 20 seconds.

However, in the bound model, the administrator has to make changes to DNS to ensure that the namespace for the failed datacenter now points to the VIP of the datacenter that is still accessible.

## Regional Namespaces

It's not uncommon, especially in larger environments, to see a combination of both the bound and unbound namespace being used.

Consider the following scenario: you are working in a multi-national company which has several datacenters across the globe. Exchange Server 2013 is deployed in both Europe and Asia Pacific. Each region also has two datacenters in which Exchange is hosted (regional resiliency).

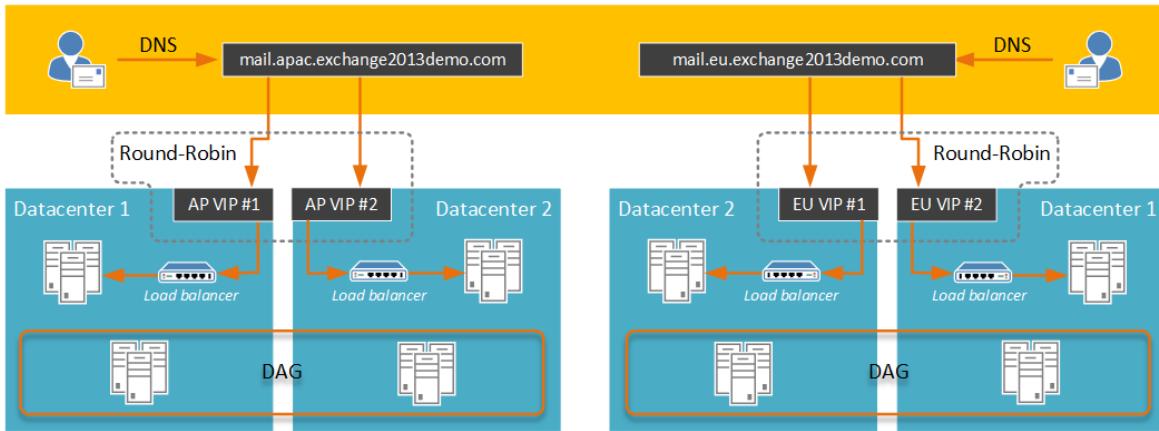
If your company wants to make sure that APAC users hit the APAC datacenters first and the same for the EU users, you should combine the bound and the unbound namespace model.

First, you create a regional namespace for both the EU and the APAC regions. For example:

- mail.eu.exchange2013demo.com
- mail.apac.exchange2013demo.com

This will ensure that users from the EU resolve the EU namespace and users in APAC will use the APAC namespace.

Next, you use the unbound model to divide traffic within the region:



In the scenario above, the DNS service will provide a request coming from Europe (using mail.eu.exchange2013demo.com) with a list of IP address for the European datacenters. Using DNS Round-Robin, the client will then connect to one of the EU datacenters.

The Client Access server infrastructure in the EU datacenters will automatically proxy traffic to the corresponding Mailbox server that hosts the user's mailbox – regardless of which EU datacenter the mailbox database is active.

The benefit of this approach is that APAC users, who might be travelling in Europe from time to time, will still receive IP addresses for the APAC datacenters. This ensures they connect to their preferred datacenters first. This is a preferred solution for some organizations than Geo DNS.

# Configuring Client Access Servers

Now that we've looked at the different ways of designing Client Access server high availability it is helpful to look at the actual steps involved in configuring Client Access servers.

For this example we'll demonstrate how to configure the following services:

- Outlook Anywhere
- Outlook Web App
- Exchange Control Panel
- Exchange ActiveSync
- Exchange Web Services
- Offline Address Book
- AutoDiscover

The namespace model being used is the unbound model, with a single HTTP namespace of "mail.exchange2013demo.com" used for all of the services. This means that the URLs being configured on the Client Access server virtual directories will be:

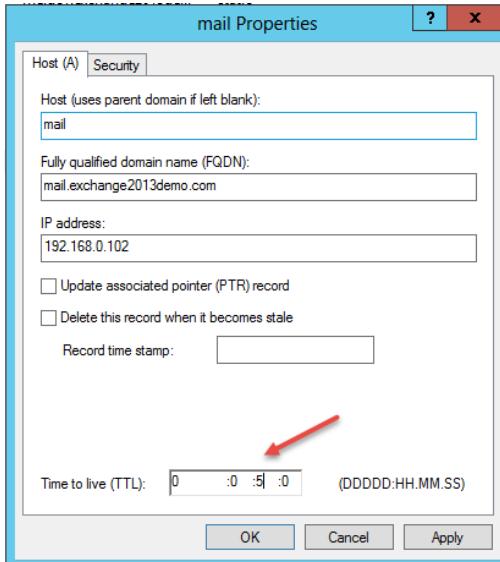
- Outlook Anywhere - mail.exchange2013demo.com
- Outlook Web App - https://mail.exchange2013demo.com/owa
- Exchange Control Panel - https://mail.exchange2013demo.com/ecp
- Exchange ActiveSync - https://mail.exchange2013demo.com/Microsoft-Server-ActiveSync
- Exchange Web Services - https://mail.exchange2013demo.com/EWS/Exchange.asmx
- Offline Address Book - https://mail.exchange2013demo.com/OAB
- AutoDiscover - https://mail.exchange2013demo.com/Autodiscover/Autodiscover.xml

## Configuring DNS Records

The namespace needs to be added to DNS so that clients can resolve the name. Because we're using DNS round robin and split DNS the records need to be added to the internal DNS servers with an A record for each Client Access server IP address.

Name	Type	Data	Timestamp
_msdcs	Start of Authority (SOA)	[521], syddc1.exchange20...	static
_sites	Name Server (NS)	meldc1.exchange2013de...	static
_tcp	Name Server (NS)	syddc1.exchange2013dem...	static
_udp	Host (A)	192.168.1.100	12/08/2014 1:00:00 AM
DomainDnsZones	Host (A)	192.168.0.100	27/10/2014 9:00:00 PM
ForestDnsZones	DAG1	192.168.0.103	27/10/2014 9:00:00 PM
(same as parent folder)	edge1	192.168.0.105	static
(same as parent folder)	edge2	192.168.0.106	static
(same as parent folder)	mail	192.168.0.102	static
(same as parent folder)	mail	192.168.0.101	static
(same as parent folder)	mail	192.168.1.102	static
(same as parent folder)	mail	192.168.1.101	static
MFTDC1	Host (A)	192.168.1.100	static

The internal DNS records should also be configured with a low TTL value so that any DNS changes required during maintenance or outage periods will take effect faster.



Once these DNS records are in place any internal client should be able to resolve the hostname and see all of the Client Access server IP addresses in the result.

```
PS C:\> nslookup mail.exchange2013demo.com
Server: localhost
Address: ::1

Name:      mail.exchange2013demo.com
Addresses: 192.168.0.102
          192.168.0.101
          192.168.1.102
          192.168.1.101
```

The name must also be added to the public (or external) DNS zone so that it can be resolved by external devices and clients. When you test this you'll need to run the DNS query from an externally connected device, or manually target the query to external DNS servers. With a single public IP address the results would look something like this.

```
PS C:\> nslookup mail.exchange2013demo.com 8.8.8.8
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Non-authoritative answer:
Name:      mail.exchange2013demo.com
Address: 58.7.236.137
```

## Configuring Virtual Directories

Although some of the hostnames and URLs are configurable using the Exchange Admin Center, some others require you to use PowerShell. So for the sake of simplicity we will use PowerShell to configure all of the services.

Because this is a high availability deployment with more than one server, and each server being configured with the same namespaces, we'll be piping commands such as Get-OWAVirtualDirectory into other commands to make the change to multiple servers at the same time.

If you were configuring multiple servers with different namespaces, such as when using the bound model or regional namespaces, then you would need to use the –Server switch to ensure you were targeting specific servers.

### *Outlook Anywhere*

To configure the internal and external host names use Set-OutlookAnywhere.

```
[PS] C:\>Get-OutlookAnywhere | Set-OutlookAnywhere -ExternalHostname mail.exchange2013demo.com -InternalHostname mail.exchange2013demo.com -ExternalClientsRequireSsl $true -InternalClientsRequireSsl $true -DefaultAuthenticationMethod NTLM
```

Note that in addition to setting the host names you must also explicitly set the SSL requirement for both internal and external clients.

The default setting for internal is False, which is fine, but we are enforcing it in this example. You also need to explicitly set either a default authentication method or an external authentication method, which has been set to NTLM in this example.

### *Outlook Web App*

To configure the OWA URLs use Set-OWAVirtualDirectory.

```
[PS] C:\>Get-OwaVirtualDirectory | Set-OwaVirtualDirectory -ExternalUrl https://mail.exchange2013demo.com/owa -InternalUrl https://mail.exchange2013demo.com/owa
```

WARNING: You've changed the InternalURL or ExternalURL for the OWA virtual directory. Please make the same change for the ECP virtual directory in the same website.

### *Exchange Control Panel*

As you can see when configuring the OWA URLs the ECP URLs must be configured to match. To configure the ECP URLs use the Set-ECPVirtualDirectory cmdlet.

```
[PS] C:\>Get-EcpVirtualDirectory | Set-EcpVirtualDirectory -ExternalUrl https://mail.exchange2013demo.com/ecp -InternalUrl https://mail.exchange2013demo.com/ecp
```

An IIS reset is required for the OWA and ECP changes to take effect, but you may prefer to wait until all of the planned changes are made before you perform the reset.

#### *Exchange ActiveSync*

To configure the ActiveSync URLs use Set-ActiveSyncVirtualDirectory.

```
[PS] C:\>Get-ActiveSyncVirtualDirectory | Set-ActiveSyncVirtualDirectory -ExternalUrl  
https://mail.exchange2013demo.com/Microsoft-Server-ActiveSync -InternalUrl  
https://mail.exchange2013demo.com/Microsoft-Server-ActiveSync
```

#### *Exchange Web Services*

To configure the EWS URLs use Set-WebServicesVirtualDirectory.

```
[PS] C:\>Get-WebServicesVirtualDirectory | Set-WebServicesVirtualDirectory -ExternalUrl  
https://mail.exchange2013demo.com/EWS/Exchange.asmx -InternalUrl  
https://mail.exchange2013demo.com/EWS/Exchange.asmx
```

#### *Offline Address Book*

To configure the OAB URLs use Set-OABVirtualDirectory.

```
[PS] C:\>Get-OabVirtualDirectory | Set-OabVirtualDirectory -ExternalUrl  
https://mail.exchange2013demo.com/OAB -InternalUrl https://mail.exchange2013demo.com/OAB
```

#### *AutoDiscover*

The final configuration is the AutoDiscover service connection point. Unlike the other host names and URLs this is not configured on a virtual directory (don't be fooled by the URLs shown when you run Get-AutoDiscoverVirtualDirectory).

To configure the new URI use Set-ClientAccessServer.

```
[PS] C:\>Get-ClientAccessServer | Set-ClientAccessServer -AutoDiscoverServiceInternalUri  
https://mail.exchange2013demo.com/Autodiscover/Autodiscover.xml
```

## Configuring an SSL Certificate

Any client that is connecting to the Client Access server namespaces will display an SSL warning to the end user due to the default self-signed certificate that is installed by Exchange setup.

So after the namespaces have been configured the server needs an SSL certificate installed that:

- Is from a trusted root certification authority
- Has each of the Client Access server names included
- Is within the valid date range (i.e., has not expired)

#### *Generate a Certificate Request*

In this example the namespaces required for the SSL certificate are:

- mail.exchange2013demo.com
- autodiscover.exchange2013demo.com
- exchange2013demo.com

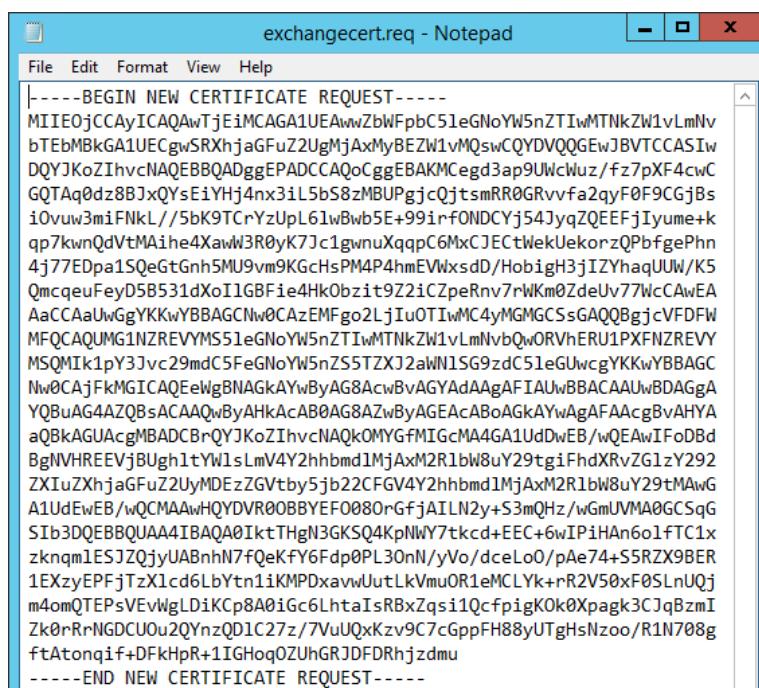
This includes the namespace that was configured for the HTTP services above, as well as the default Autodiscover names that non-domain members such as mobile devices will attempt to use during email account setup.

To generate the certificate request on the first Client Access server we use New-ExchangeCertificate.

```
[PS] C:\>$Data = New-ExchangeCertificate -GenerateRequest -SubjectName "c=AU, o=Exchange 2013 Demo, cn=mail.exchange2013demo.com" -DomainName mail.exchange2013demo.com,autodiscover.exchange2013demo.com,exchange2013demo.com -PrivateKeyExportable $true

[PS] C:\>Set-Content -Path "C:\temp\exchangepcert.req" -Value $Data
```

The result is a text file containing the certificate request data.



```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIE0jCCAYICAQAwTjEiMCAGA1UEAwZbWFpbC5leGNoYW5nZTIwMTNkZW1vLmNvbTEbMBkGA1UECgwSRXhjaGFuZ2UgMjAxMyBEZw1vMQswCQYDVQQGEwJBVTCCASIwDQYJKoZIhvCNQAEQQADggEPADCCAQoCggEBAKMCegd3ap9UWcWuz/fz7pXF4cwCGQTAq0dz8BjxQysE1YH4nx3l5bS8zMBUPgjcQjtsmRR0Grvvfa2qyF0F9CGjBsioVuw3m1FNkL//5bK9TCrYzUpL61wBwb5E+991rfONDcyj54JyqZQEEFjIyume+kqp7kwnQdVtMAihe4XawW3R0yK7Jc1gwnuXqqpC6MxCJEctWekUekorZPbfgePhn4j77EDpa1S0eGtGnh5MU9vm9KGchSPM4P4hmEVwxsD/Hobigh3jIZYhaqUW/K5QmcceuFeyD5B531dXoI1GBFie4HkObzit9Z2icZpeRnv7wKm0ZdeUv77WccAwEAaCCAAuWggyKKWYBBAGCNw0CAzEMFgo2LjIu0TiwMc4yMGMGCSsGAQ0BjgcVFDFWMFQCAQUMG1NZREVYMS5leGNoYW5nZTIwMTNkZW1vLmNvbQwORVhERU1PXFNZREVM5QMIk1pY3Jvc29mdC5FeGnoYW5nZ5TZXJ2aNN1SGzdC51eGUwcgYKKwYBAGCNw0CajFKMGICAQEEwlBNAGkAYwByAG8AcwBvAGYAdAAgAFIAUwBACAAUwBDAGgAYQ8uAG4AZQBsACAAQwByAHkAcAB0AG8AZwByAGEAcABoAGkAYwAgAFAAcgBvAHYAaQBkAGUAcgmBADCBrQYJKoZIhvCNQkOMYGFMIgCMA4GA1UdDwEB/wQEAwIFoDBdBgNVHREEVjbUghltYwlsLmV4Y2hhbmd1MjAxM2R1bW8uY29tgiFhdXRvZG1zY292ZXIUZXhjaGFuZ2UyMDEzZGVtby5jb22CFGV4Y2hhbmd1MjAxM2R1bW8uY29tMaWGA1UdEwEB/wQCMAAwHQYDVR00BYEF0080rGfjA1LN2y+S3mQHz/wGmUVMA0GCSq6SIB3DQE8BQUAA4IBAQAOIkTHgN3GKSQ4KpNW7tkcd+EEC+6wIP1HAn6o1FTC1xzknqm1ESJZQjyUABnhN7fQeKFY6Fdpl30nN/yVo/dceLoO/pAe74+S5RZX9BER1EXzyEPFjTzX1cd6LbYtn1iKMPDxavwUutLkVm0R1eMCLYk+rR2V50xF0SLnUQjm4omQTEPsEVwgLD1KCp8A0iGc6LhtaIsRBxZqs1QcfpigK0k0Xpagk3CJqbzmIZk0RrNGDCU0u2QYnzQD1C27z/7VuUQxKzv9C7cGppFH88yUTgHsNzoo/R1N708gftAtonqif+DFkHpR+1IGHqoZUhGRJDfDRhjzdmu
-----END NEW CERTIFICATE REQUEST-----
```

### *Submit the Certificate Request to a Certification Authority*

The certificate request can now be submitted to a certification authority. The exact steps for this will vary depending on the company you choose, but will generally be:

1. Submit the certificate request file, or copy/paste the contents into a form
2. Provide payment details
3. Follow any additional approval or verification steps that they require
4. Download the issued certificate file

If you are completing this process for a test or training environment you can use your own private CA as a way to save on costs.

- [How to Issue an SSL Certificate for Exchange Server 2013 from a Private Certificate Authority](#)<sup>2</sup>

### *Complete the Pending Certificate Request*

When you have received the certificate, which will likely be provided as a file with the extension .cer, the pending certificate request can be completed using Import-ExchangeCertificate.

```
[PS] C:\>Import-ExchangeCertificate -FileData ([Byte[]]$([Get-Content -Path c:\temp\certnew.cer -Encoding byte -ReadCount 0))  
Thumbprint Services Subject  
----- -----  
F11A38611BE4DF09A25F25A07FFF054C4E808919 ..... CN=mail.exchange2013demo.com, O=Exchange 2013  
Demo, C=AU
```

### *Export/Import the SSL Certificate to Multiple Exchange Servers*

The same SSL certificate should be used on all of the Client Access servers that are being used for the same namespace. Now that the certificate has been installed on the first server we can export it and import it to the other servers.

To export the SSL certificate use Export-ExchangeCertificate. A UNC path is recommended for the export location.

```
[PS] C:\>$file = Export-ExchangeCertificate -Thumbprint F11A38611BE4DF09A25F25A07FFF054C4E808919 -BinaryEncoded:$true -Password (Get-Credential).Password  
[PS] C:\>Set-Content -Path \\sydex1\c$\temp\exchangecert.pfx -Value $file.FileData -Encoding Byte
```

Note that the thumbprint value is the same as was displayed to you when you completed the pending certificate request in the last section. Also, when the dialog box appears to prompt you for credentials, you must enter a username (any username, it doesn't matter) but only the password is used. You must remember the password for the next step of importing the certificate.

<sup>2</sup> <http://exchangeserverpro.com/exchange-2013-ssl-certificate-private-certificate-authority/>

To import the SSL certificate to other servers use Import-ExchangeCertificate.

```
[PS] C:\>Import-ExchangeCertificate -Server SYDEX2 -FileData ([Byte[]]$([Get-Content -Path \\sydex1\c$\temp\exchangecert.pfx -Encoding byte -ReadCount 0])) -Password:(Get-Credential).password

Thumbprint          Services   Subject
-----            -----      -----
F11A38611BE4DF09A25F25A07FFF054C4E808919 ..... CN=mail.exchange2013demo.com, O=Exchange 2013
Demo, C=AU

[PS] C:\>Import-ExchangeCertificate -Server MELEX1 -FileData ([Byte[]]$([Get-Content -Path \\sydex1\c$\temp\exchangecert.pfx -Encoding byte -ReadCount 0])) -Password:(Get-Credential).password

Thumbprint          Services   Subject
-----            -----      -----
F11A38611BE4DF09A25F25A07FFF054C4E808919 ..... CN=mail.exchange2013demo.com, O=Exchange 2013
Demo, C=AU

[PS] C:\>Import-ExchangeCertificate -Server MELEX2 -FileData ([Byte[]]$([Get-Content -Path \\sydex1\c$\temp\exchangecert.pfx -Encoding byte -ReadCount 0])) -Password:(Get-Credential).password

Thumbprint          Services   Subject
-----            -----      -----
F11A38611BE4DF09A25F25A07FFF054C4E808919 ..... CN=mail.exchange2013demo.com, O=Exchange 2013
Demo, C=AU
```

### *Assign the SSL Certificate to Exchange Services*

Finally we can enable the SSL certificate for services on the Exchange server.

Notice in the previous step that the thumbprint for the certificate remained the same as it was imported to each server. We can use Enable-ExchangeCertificate to enable the SSL cert on each server.

```
[PS] C:\>Enable-ExchangeCertificate -Thumbprint F11A38611BE4DF09A25F25A07FFF054C4E808919 -Server SYDEX1
-Services IIS

[PS] C:\>Enable-ExchangeCertificate -Thumbprint F11A38611BE4DF09A25F25A07FFF054C4E808919 -Server SYDEX2
-Services IIS

[PS] C:\>Enable-ExchangeCertificate -Thumbprint F11A38611BE4DF09A25F25A07FFF054C4E808919 -Server MELEX1
-Services IIS

[PS] C:\>Enable-ExchangeCertificate -Thumbprint F11A38611BE4DF09A25F25A07FFF054C4E808919 -Server MELEX2
-Services IIS
```

# Client Access Server Summary

High availability for Client Access services in Exchange Server 2013 involves the deployment of multiple, load balanced servers.

The important decisions you need to make are the namespace model you are going to deploy, which then feeds into the SSL certificate requirements. You can then determine which load balancing method you are going to deploy to align with your namespace model.

# Mailbox Server High Availability

The high availability feature for Exchange Server 2013 Mailbox servers is the Database Availability Group.

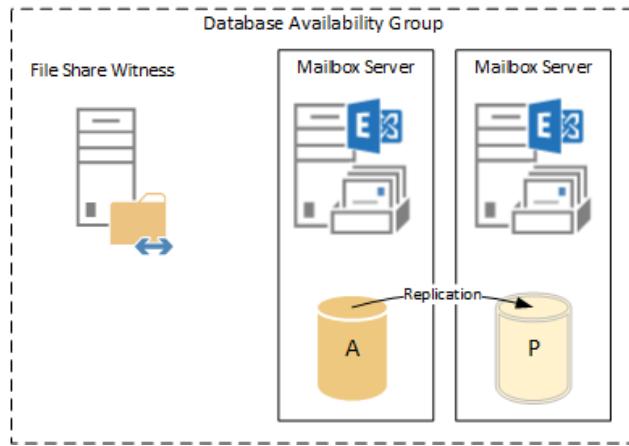
Exchange 2013 Database Availability Groups (DAGs) are very similar to Exchange 2010 DAGs, but also deliver a series of improvements and new features for customers. If you're not already familiar with Exchange 2010 DAGs don't worry, we won't be assuming any prior knowledge or experience as we go through the topic in this chapter.

## Overview of Exchange Server 2013 Database Availability Groups

A Database Availability Group (DAG) consists of up to 16 Exchange 2013 Mailbox servers, and one or more additional servers (that can be a non-Exchange server if necessary) that may be required to act as a File Share Witness (FSW - more on this shortly).

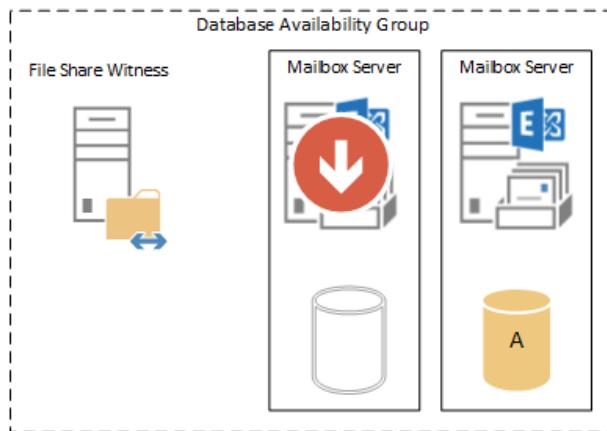
The Mailbox servers within a DAG are capable of hosting a copy of a mailbox database from another DAG member; up to the Exchange 2013 limit of 100 mailbox databases per server (that includes both active and passive database copies).

A simple example of a Database Availability Group would be as follows.



In the example above a Database Availability Group (DAG) has been created with two mailbox servers as DAG members. A single database is active on the first server, and replicating changes to the passive copy on the second server. An additional server is performing the role of the File Share Witness (FSW).

This deployment can provide high availability for the mailbox database because when a single server goes offline the database copy on the other server can be made active and brought online to continue serving requests for mailbox data.



Of course such a simple looking capability happens to also be quite complex behind the scenes, especially as the DAG grows to the maximum possible size of 16 members, each hosting up to 100 active or passive database copies, and possibly even spanning multiple physical locations within your environment.

But DAGs are also fairly easy to understand once you learn the fundamentals. Let's take a look at some of those fundamental concepts now that we'll be diving deeper into as we go through this chapter.

The following terminology will be used frequently throughout this chapter. You may need to refer back to this list from time to time as we cover the topic of Mailbox server high availability.

## DAG Members

The Mailbox servers that make up a database availability group are referred to as DAG members.

There can be up to 16 members of a DAG, and there are a few things to be aware of when it comes to servers being eligible to be DAG members.

- DAG members must be running the same version of Exchange Server. This means that DAG members can't be a mix of Exchange 2010 and Exchange 2013 servers. However, a DAG can be a mix of different Exchange Server 2013 Cumulative Update or Service Pack build levels. This scenario will naturally occur as you are upgrading DAG members with the latest CU or Service Pack, and it is fully supported, however it is also recommended to upgrade all DAG members to the same build in a timely manner.
- DAG members must be running the same version of the Windows Server operating system. For example, all DAG members can run Windows Server 2012, but can't run a mixture of Windows Server 2012 and Windows Server 2012 R2. This is because the DAG uses an underlying Windows Failover Cluster, and it is not supported (or even possible) to run a Windows Failover Cluster with different versions of Windows Server.
- The dependency on Windows Failover Clustering also means that Exchange Server 2013 DAG members running on Windows Server 2008 R2 must be installed with Enterprise or Datacenter Edition for the operating system, as Standard Edition does not include Windows Failover Clustering.
- For DAG members running Windows Server 2012 or 2012 R2, either Standard or Datacenter Edition can be chosen, as both support Windows Failover Clustering.
- DAG members can run either Standard or Enterprise Edition of Exchange Server 2013. The only difference between the two editions is the number of database copies the Exchange server can host (5 for Standard Edition, 100 for Enterprise Edition).

## Incremental Deployment

In earlier versions of Exchange Server (2007 and earlier) if a clustered Exchange server was being deployed you were required to form the underlying cluster first, then install Exchange Server onto the cluster using special setup steps that were different to a standalone server install.

This meant that if you later needed to change from a standalone server to a clustered server, you had to build entirely new servers and migrate mailboxes to the cluster, then decommission the standalone server.

With database availability groups we avoid that problem thanks to the concept of incremental deployment. Incremental deployment allows us to take a standalone Exchange Server 2013 Mailbox server and form a DAG without impacting the databases and mailboxes already hosted on the server. No need to build the cluster first, then install Exchange on top afterwards.

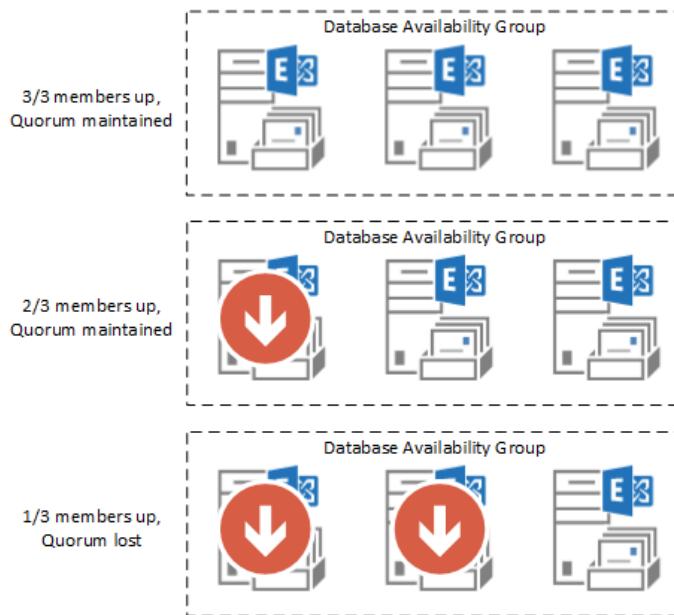
## Quorum

An Exchange Server 2013 DAG utilizes Windows Failover Clustering and the concept of quorum. This underlying cluster is managed automatically for you by Exchange, so you don't need to worry about it much other than to be aware of how quorum works.

If the concept of quorum is new to you just think of it as a voting process in which a majority of voting members must be present to make a decision. The decision in the case of a DAG is basically whether the DAG (and all of the databases in that DAG) should be online or offline.

Because a majority of votes is required for quorum there are two different quorum models used depending on how many DAG members you have.

For a DAG with an odd number of members the Node Majority quorum mode is used.

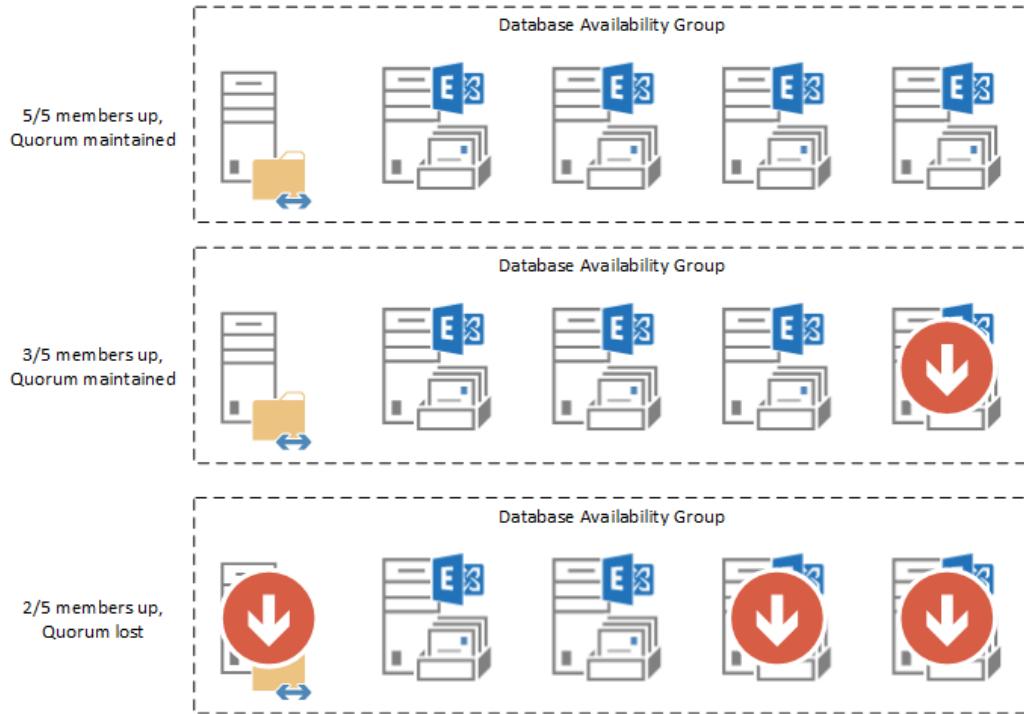


In the above example a three member DAG is able to maintain quorum during a single server failure, but quorum is lost when two servers are unavailable.

For a DAG with an even number of members the Node and File Share Majority quorum mode is used. This mode involves an additional server referred to as the File Share Witness (FSW).

The FSW is typically another Exchange server located in the same site as the DAG members, so that it is a server under the administrative control of the Exchange administrators. While it cannot be a member of the same DAG for which it is also performing the FSW role, it can be a member of another DAG. In other words, if you had two DAGs in your organization, one of the members of the first DAG could perform the FSW role for the second DAG, and vice versa.

However the FSW can also be any Windows server (such as a file server) if another Exchange server doesn't exist.



In the above example a four member DAG is using an additional server as the FSW. The DAG is able to maintain quorum with up to two server failures, but quorum is lost when three servers are down.

**Note:** The File Share Witness can be unavailable for periods of time, such as during installation of routine security updates, without impacting the DAG, as long as the DAG members are still able to form quorum.

As you can see quorum is an important concept to understand when it comes to database availability groups and Mailbox server high availability.

## Dynamic Quorum

DAGs deployed on Windows Server 2012 or 2012 R2 can be more resilient to multiple node failures thanks to a feature called dynamic quorum.

Dynamic quorum is enabled by default. In dynamic quorum the cluster dynamically manages the assignment of votes to cluster nodes (the DAG members), based on the current state of each node.

The cluster will automatically remove votes from nodes that no longer have active cluster membership (i.e. when they are offline). The vote is automatically assigned again when a node rejoins the cluster (i.e. comes back online).

Dynamic quorum also makes it possible, under the right circumstances, for a cluster (or more to the point, a DAG) to stay online even with just one last surviving cluster node. This is achieved by dynamically adjusting the quorum majority requirement as nodes become unavailable.

Does this mean we can be less concerned about maintaining quorum for our DAG by ensuring that a majority of DAG members remain online? Not really. Here's what Microsoft has to say about dynamic quorum and Exchange Server 2013 DAGs<sup>3</sup>.

- Dynamic quorum does not change quorum requirements for DAGs
- Dynamic quorum does work with DAGs
- All internal (Microsoft) testing is performed with dynamic quorum enabled (in other words, they do not test scenarios with dynamic quorum disabled, so you should not disable it)
- Dynamic quorum is enabled for DAG members in Office 365
- Exchange is not dynamic quorum-aware (in other words, Exchange has no visibility of what the underlying cluster is doing in terms of dynamic quorum)
- Leave it enabled
- Don't factor it into availability plans (in other words, don't rely on it)

**Real World:** The “last man standing” terminology sometimes used to describe dynamic quorum is a little misleading, because there are scenarios where the last server still online is unable to form quorum even with dynamic quorum enabled. But let's face it, a situation where your DAG has suffered multiple failures and is down to one server online is going to be quite rare, and presents a problem much larger than whether dynamic quorum was able to save the day or not.

## File Share Witness

We've already mentioned the file share witness while discussing the concept of quorum in an earlier section of this chapter. But let's dive a little deeper into exactly what the file share witness is, and what it does in the DAG.

The FSW is effectively another vote holder in the DAG that acts as a tie-breaker when the DAG has an even number of members. For example, a four member DAG with one failed member can form quorum with the remaining three members. However if a second member goes offline, with only two out of four members online a majority of nodes is no longer available to form quorum, and the FSW is needed to form a majority.

---

<sup>3</sup> Source: [Ignite Webcast: Exchange Server 2013 High Availability](#)

DAGs with odd numbers of members don't need the FSW to vote because quorum can be formed without it. For example, a three member DAG with one failed member can still form quorum with the remaining two members, because two out of three is still a majority.

Even if the FSW is not being actively used (i.e. in a DAG with an odd number of members) it is still defined in the DAG configuration.

The FSW can be any Windows Server, it doesn't necessarily need to be another Exchange server. However it is common for the FSW to reside on an Exchange server because the server is then in control of the Exchange administrators, and has the necessary permissions already in place.

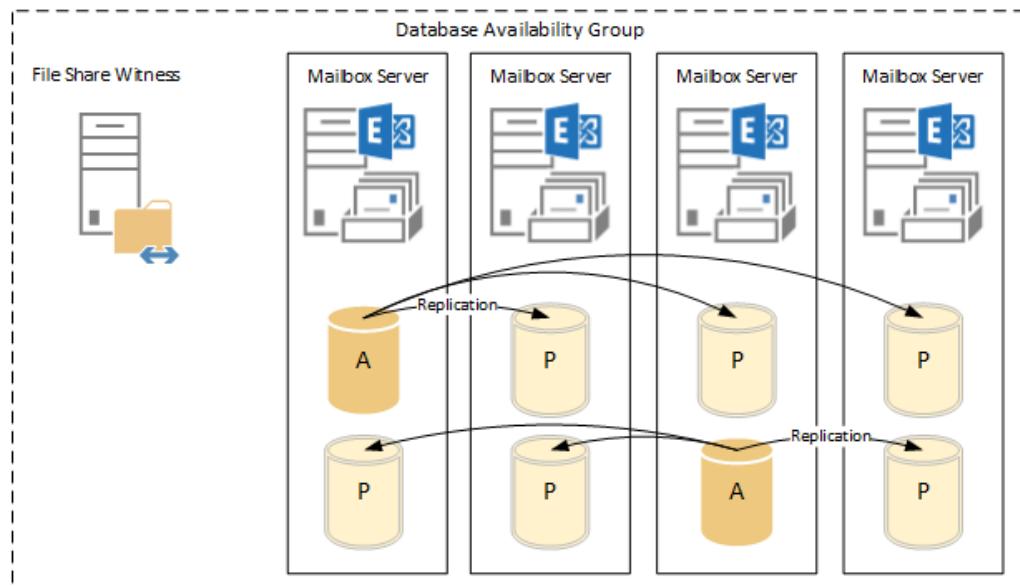
But those permissions can be configured on any Windows Server by adding the Exchange Trusted Subsystem security group into the local Administrators group of the server. The FSW can even be a domain controller, which has no local Administrators group so therefore would need the Exchange Trusted Subsystem security group added to the Domain Admins group in Active Directory. Although this is not generally recommended it is still a supported configuration, and may be the only option available for some customers.

## Active and Passive Database Copies

The active database copy is the copy of a database that is mounted and actively servicing clients. There can be only one active copy of a database at any given time.

The passive database copy is a copy of a database that is dismounted and is receiving changes through continuous replication from the active database copy. As the maximum number of DAG members is 16 there can be up to 15 passive copies of a single database.

Within a DAG that has multiple members hosting multiple databases, each DAG member can host a mixture of active and passive database copies.



DAG members running Exchange Server 2013 Standard Edition can host up to a total of 5 active or passive database copies, while Enterprise Edition allows for up to a total of 100 active or passive database copies to be hosted by a DAG member.

## Continuous Replication

Each DAG member hosting a copy of a given mailbox database participates in a process of continuous replication to keep the copies consistent.

Database replication occurs between Exchange Server 2013 DAG members using two different methods:

- File mode replication
- Block mode replication

During File Mode replication each transaction log is fully written (a 1MB log file) and then copied from the DAG member hosting the active database copy to each DAG member that hosts a passive copy of that database.

The other DAG members then replay the transaction log file into their own passive copy of the database to update it.

File mode replication has an obvious downside in that a transaction log that hasn't already been copied to the other DAG members may be lost if the DAG member hosting the active database copy suffers a log storage failure or becomes permanently unavailable.

Although there are other recovery mechanisms to minimize the impact of this scenario, this is a reason why file mode replication is used only during the initial seeding of a database copy. After seeding is complete the database switches automatically to block mode replication.

During block mode replication as each database transaction is written to the log buffer on the active server it is also sent to the log buffer of DAG members hosting passive copies of the database.

As the log buffer becomes full each member of the DAG is then able to build their own transaction log file for replay into their own database copy.

Block mode replication has advantages compared to file mode replication when there is a failure in the DAG, because less transaction log data is likely to be lost.

In addition to the two modes of operation, you can also suspend and resume database replication when necessary, for example when performing maintenance on a DAG member. We'll look closer at this in a later section on managing database copies.

## Lagged Database Copies

A lagged database copy is a passive database copy that has a delayed log replay time configured.

Normally a passive database copy will replay the transaction log data into the database immediately, so that the passive database copy is as up to date as possible.

With a lagged database copy the administrator sets a delay on the log replay, so that the database copy “lags” behind the others in terms of the latest updates. This lag interval is often several days, and allows the administrators to use the lagged database copy for some recovery scenarios.

We'll look closer at configuring and using lagged database copies later in this chapter.

## Active Manager

Active Manager is a component of Exchange that is responsible for monitoring the status of the database availability group, detecting failures, and making decisions about corrective actions that should be taken.

On DAG members there are two types of Active Manager; Primary and Standby.

The Primary Active Manager tracks the topology of the DAG for issues such as server failures, and makes the decisions about which database copies are active and passive. For example, the Primary Active Manager will detect if an active database copy has gone offline and initiate corrective action to failover to another database copy.

The Standby Active Manager(s) monitors its own local database copies and notifies the Primary Active Manager of any issues that it detects.

## Switchovers and Failovers

Switchovers and failovers occur when the active copy of a database moves from one DAG member to another.

Remember that databases are continuously replicating to other DAG members that host a copy of that database. So when we say the active database “moves”, it is not actually the database files themselves that move. Instead, it is a process of dismounting the copy of the database on the DAG member where it is active, and mounting the copy of the database stored on another DAG member.

Switchovers are deliberate, administrator-driven events where the active database copy is moved to a different server. An example scenario where a switchover is performed would be during routine maintenance such as applying security updates to a DAG member.

The server hosting the active database copy is moved to may be specifically chosen by the administrator, or it may be chosen automatically by the DAG if for example the switchover has been initiated by the [StartDAGServerMaintenance.ps1 script](#)<sup>4</sup>.

A failover is similar to a switchover except it is usually not a deliberate, administrator-driven event. Instead it is a system response to a fault or failure scenario, such as a DAG member going offline due to hardware failure. If the DAG member hosting an active database copy is no longer online the DAG responds by choosing another DAG member to mount its passive copy of the database and become the active copy.

## Activation Preference

The activation preference is a value configured by the administrator on each copy of a database within a database availability group. The lowest value of 1 indicates the most preferred database copy. Each database copy has a unique value, for example if there are four database copies then the assigned activation preference values will be 1, 2, 3 and 4.

The key word here is *preference*. The activation preference value is one of many factors used by Active Manager during the “Best copy and server selection” process. It is not strictly adhered to as the only order in which database copies can become active in a failover scenario.

However, it can be used when running the RedistributeActiveDatabases.ps1 script to rebalance a database availability group. In other words, to move each active database copy back to the copy with activation preference of 1, assuming it is a healthy copy at the time.

**Note:** You should set the activation preference values to the order in which you prefer your database copies to become active in a failover scenario. This will allow RedistributeActiveDatabases.ps1 to be a useful script for you when rebalancing the DAG after maintenance or a fault has occurred. Just don't be surprised if the copy with AP=3 mounts before the copy with AP=2 in some failure scenarios, due to the way best copy selection works.

## Best Copy and Server Selection

When the Primary Active Manager makes a decision about which copy of a database should become the active copy it uses a process called best copy and server selection (BCSS).

Best copy and server selection is a complex process involving a number of variables that are assessed by the DAG so that a database copy can be chosen to become active to restore availability of service, while balancing that objective against the risk of data loss if a database copy that is not fully up to date with the latest changes was made active.

<sup>4</sup> <http://exchangeserverpro.com/how-to-rebalance-mailbox-databases-in-exchange-server-dag/>

These variables include:

- Database and server configurations such as AutoDatabaseMountDial and activation policies
- The health of database copies within the DAG
- The health of server components monitored by managed availability

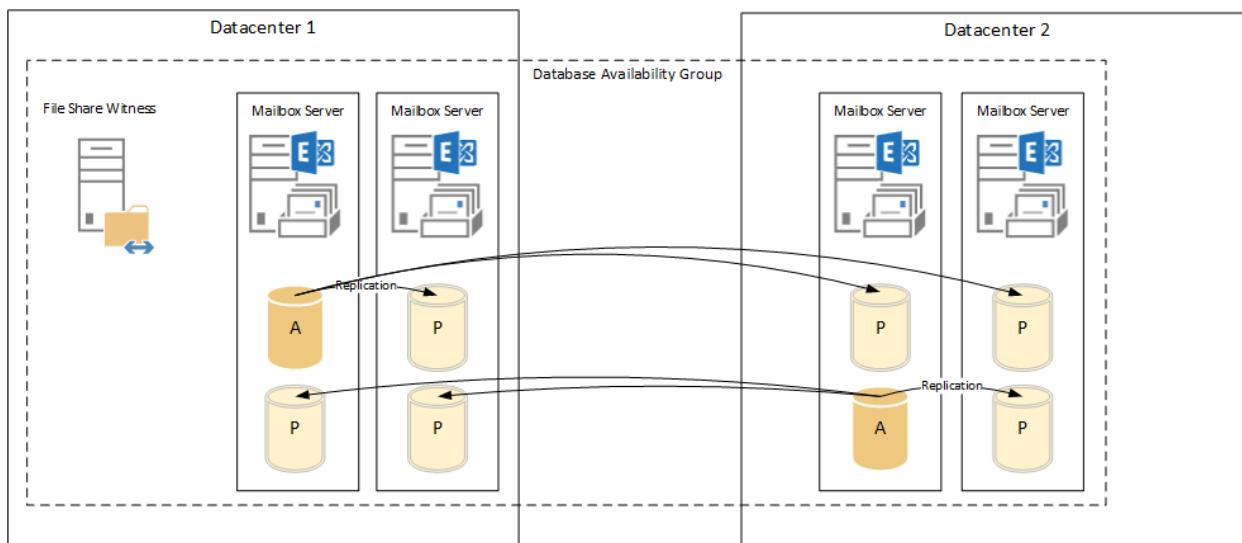
For more on BCSS see the “Best Copy Selection in Action” section later in this chapter.

## High Availability vs Site Resilience

Site resilience takes high availability to the next level by extending the concept across multiple sites (or datacenters).

Site resilience ensures that in the event of a full datacenter failure, all required services can be made available in an alternative datacenter.

In the case of Exchange Server 2013 this will involve stretching a database availability group across both sites by placing DAG members in both datacenters, and therefore hosting database copies in both sites. The continuous replication process runs across the WAN link between the two datacenters.



Site resilience can involve fully automated, semi-automated, or even fully manual recovery actions to bring services online when a datacenter has failed. Later in this chapter we'll explore a multi-site DAG configuration, as well as looking at datacenter failure scenarios.

**Real World:** The recommended practice for site resilient deployments is to configure each datacentre as a separate Active Directory Site. This enables Safety Net to correctly provide site-resilience for the transport layer, as well as adhering to Active Directory best practices for less than 10ms latency between subnets in the same Active Directory Site.

## Datacenter Activation Coordination Mode

Datacenter Activation Coordination (DAC) Mode is a property of DAGs that is designed to prevent split brain conditions from occurring by enabling a protocol called Datacenter Activation Coordination Protocol (DACP).

In addition, DAC Mode enables the use of three PowerShell cmdlets for site-resilience:

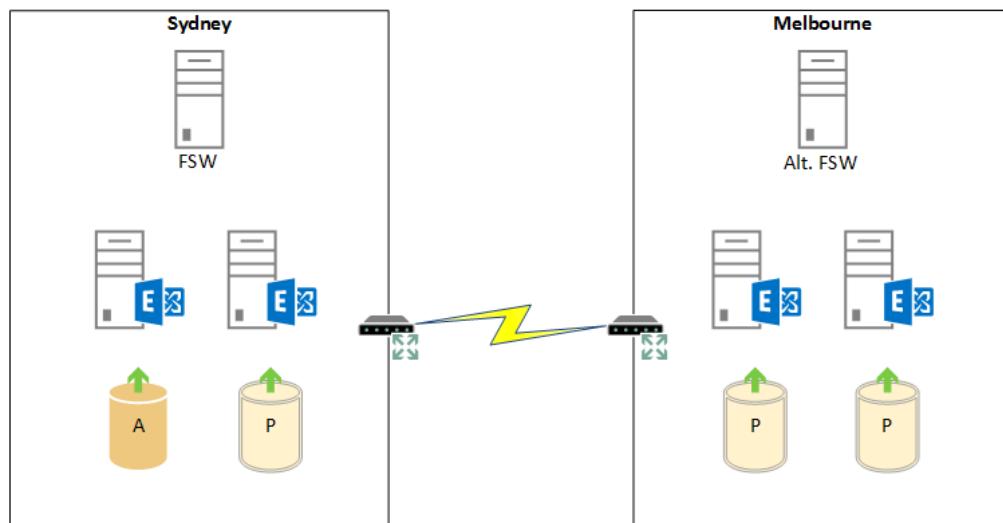
- Stop-DatabaseAvailabilityGroup
- Restore-DatabaseAvailabilityGroup
- Start-DatabaseAvailabilityGroup

Without those cmdlets any datacenter switchover or failover scenario involves using other combinations of Exchange and cluster management tools. These site resilience cmdlets make datacenter switchovers and failovers much easier to manage.

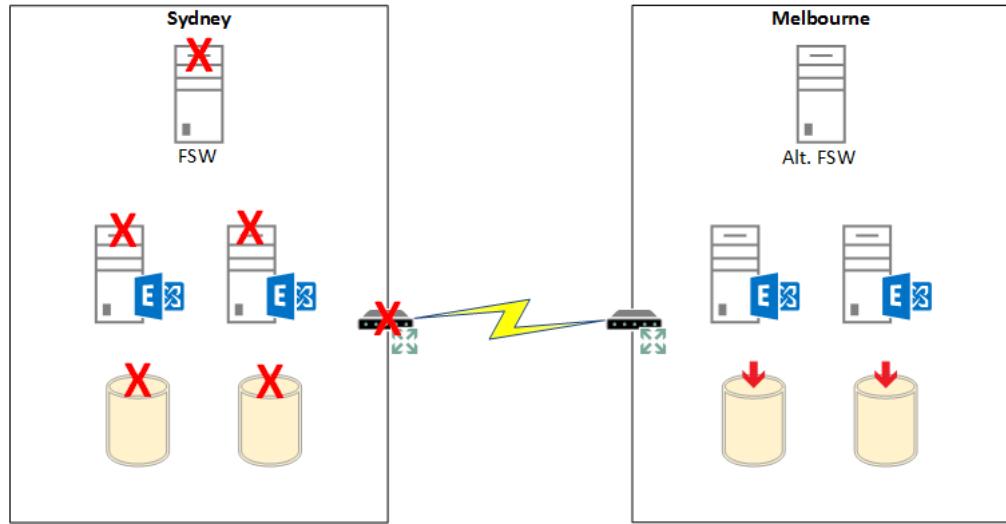
We'll look closer at the use of the site resilience cmdlets later in this chapter when we demonstrate a site failure scenario. For now let's continue discussing split brain conditions.

A split brain condition can occur in a multi-site DAG when one datacenter goes offline entirely. It can also occur in a single-site DAG in some network failure situations. Let's take a look at an example of a multi-site failure where the benefits of DAC mode become clear.

In this example the Sydney and Melbourne datacenters each host two DAG members, with Sydney also hosting the file share witness server. To keep this example simple a single database exists in the DAG, currently active on a Sydney DAG member.

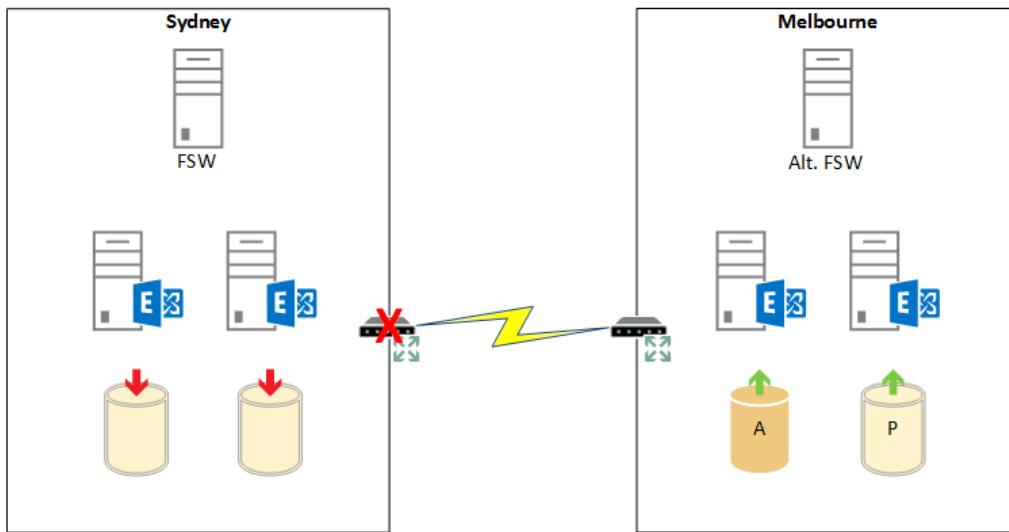


The Sydney datacenter has a power failure that takes the entire site offline. With two DAG members and the FSW offline in Sydney, and just two DAG members online in Melbourne, quorum can't be maintained and the database goes offline.

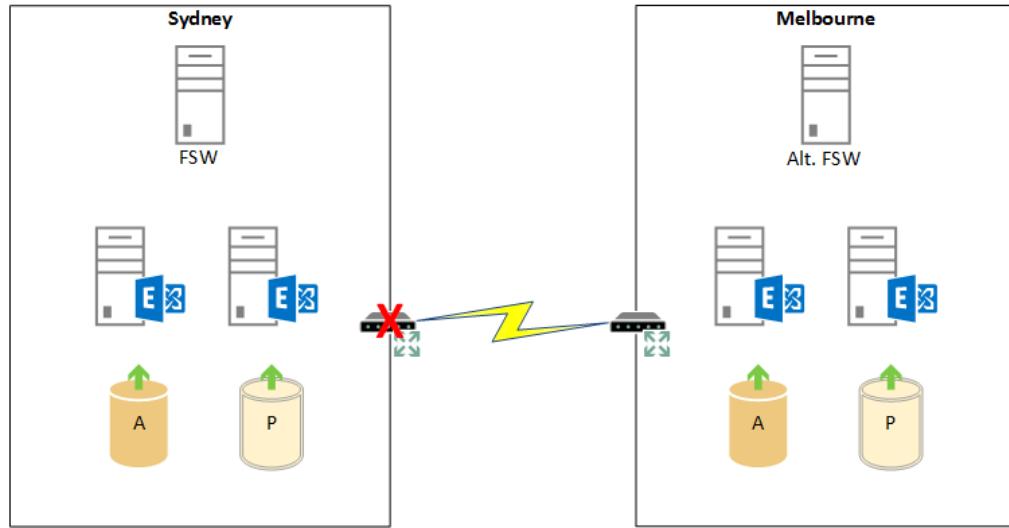


The administrators activate the alternate file share witness in Melbourne to restore quorum, and bring the database online in Melbourne to restore service.

Eventually the datacenter in Sydney has power restored and the Sydney DAG members and file share witness come back online. However, the WAN connection remains offline, preventing the DAG members in each site from communicating with each other.



The two Sydney DAG members and file share witness have enough votes to achieve quorum, so the database is brought online in Sydney.



At this stage the problem should be apparent. Both Sydney and Melbourne have an active copy of the same database because the DAG members in each site were not able to communicate with each other. A split brain condition has occurred.

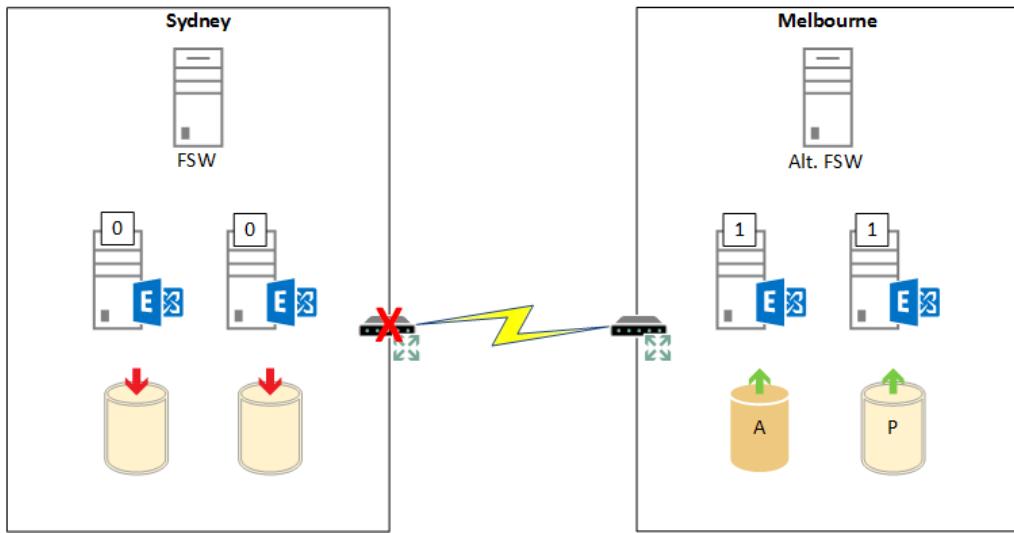
DAC and DACP prevent this behavior by requiring a DAG member to check with other DAG members before it is allowed to bring database online.

DACP exists as a bit (a 0 or 1) that is stored in memory. When DAC mode is enabled each DAG member starts up with a DACP bit of 0. Until it can communicate with a DAG member that has a DACP bit of 1, or alternatively it can communicate with every other member of the DAG, it will not attempt to activate its database copies even if it can achieve quorum with some of the DAG members.

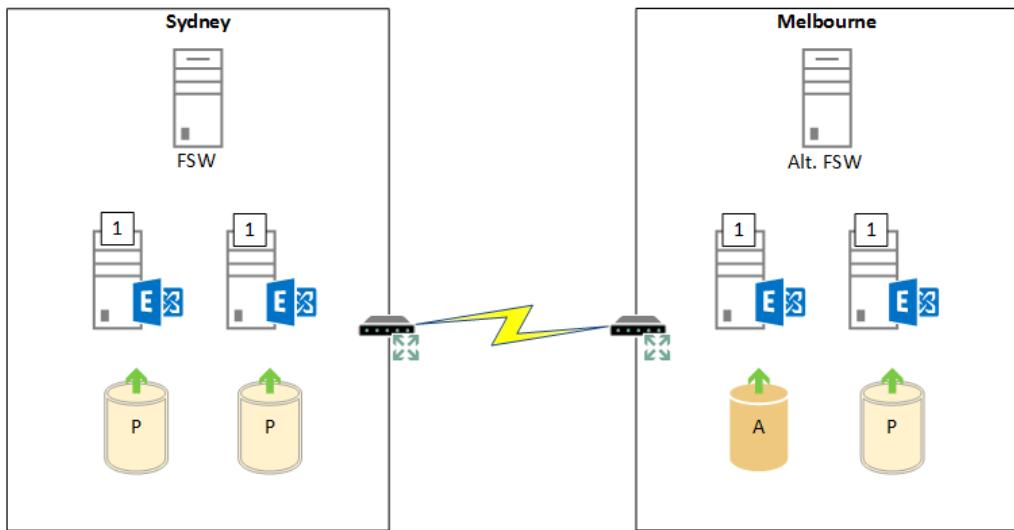
To demonstrate this let's go back in the example scenario above to the stage where the Sydney datacenter was coming back online again.

When DAC Mode has been configured in advance the Sydney DAG members start up with a DACP bit of 0 and are unable to communicate with the Melbourne DAG members because the WAN link is still offline.

Therefore they do not bring the database online in Sydney, preventing a split brain condition.

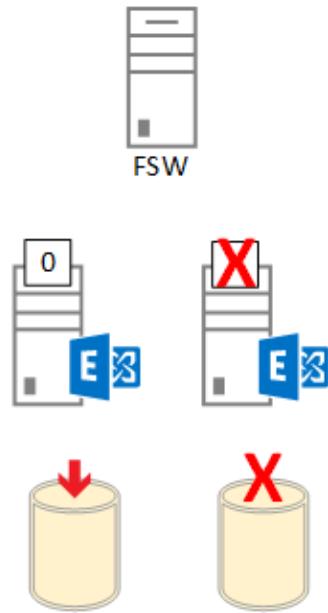


When the WAN connection is restored the Sydney DAG members are able to communicate with the Melbourne DAG members. Their DACP bit is set from 0 to 1 and, because they now realize that the database is already active in Melbourne, their database copies become passive copies.



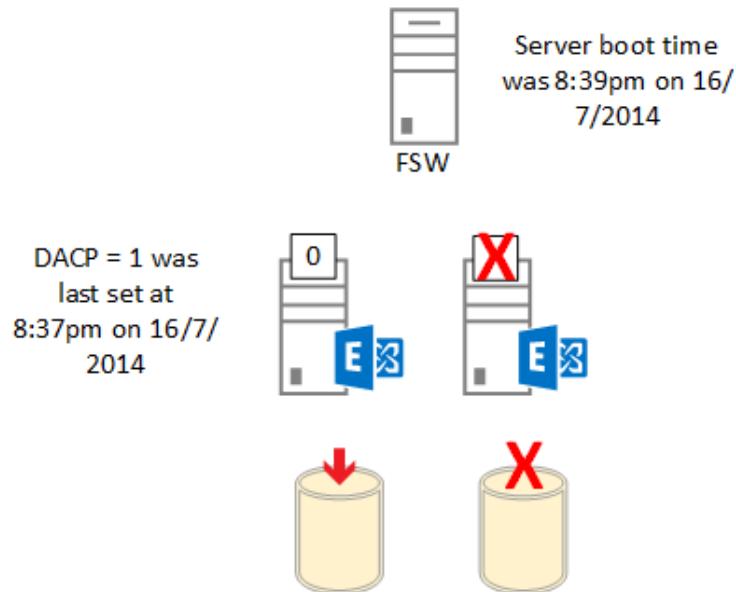
DAC mode provides the same benefits for DAGs with as few as two members, in that it prevents split brain conditions and enables the use of the site resilience PowerShell cmdlets, whether those two members are in the same or separate datacenters.

When a DAG with DAC Mode enabled has only two members, and one of those DAG members is offline due to a fault, the sole remaining DAG member comes online after a restart with a DACP bit of 0.

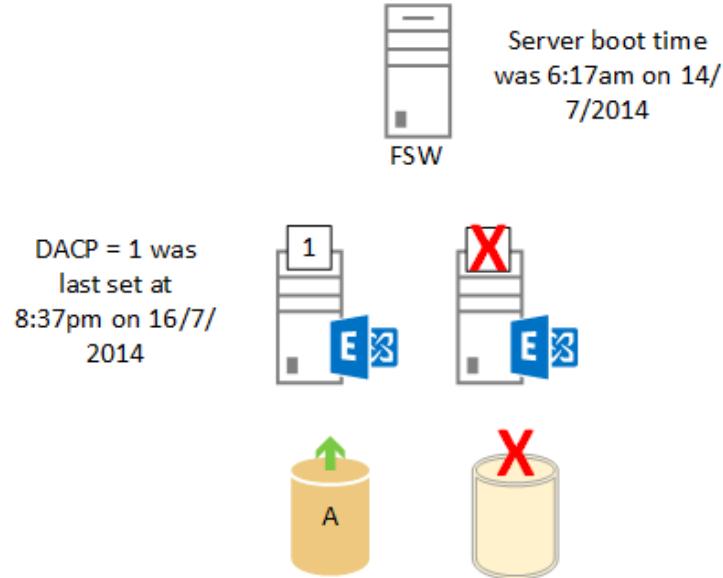


In this situation there are no other DAG members with a DACP bit of 1 that are available, so the DAG member performs an additional check against the current witness server for the DAG. The time that the DACP bit was last set to 1 is compared to the boot time of the witness server.

If the time that the DAG member's DACP bit was set to 1 was before the boot time of the witness server, the system assumes that the two servers were rebooted at the same time (which may indicate a local datacenter outage had occurred), and the DAG member will not be allowed to mount any databases.



However, if the time that the DAG member's DACP bit was set to 1 was after the boot time of the witness server, the system assumes that the DAG member rebooted on its own for other reasons such as server maintenance, and the DAG member will be allowed to mount databases.



Of course this leaves open the possibility that the system will make the wrong decision and will prevent the DAG member from mounting databases when you actually want it to mount them. You should think of this as the DAG erring on the side of caution to avoid a split brain condition, which most people would agree is the correct outcome.

In cases where you determine that the sole remaining DAG member did not automatically mount databases you can reset the DACP bit to 1 manually by running the `Restore-DatabaseAvailabilityGroup` command.

**Real World:** A split brain scenario might seem unlikely for a two member DAG where the servers are both in the same physical location, perhaps even located in the same server rack. You may even consider the possibility of DAC Mode erring on the side of caution and preventing databases from mounting to be an undesirable outcome.

However, even if a split brain is unlikely to occur it is a major disaster when it does happen. In comparison, enabling DAC mode is harmless and only delivers benefits, so the recommendation is to enable it for DAGs with two or more members. The exception is DAGs that use third-party replication, unless the replication vendor has specified that DAC mode should be enabled.

## Related Transport Features

There are scenarios with an Exchange Server 2013 DAG where an out of date copy of a mailbox database is mounted. For example, consider a scenario in which a database copy that was still missing some log files when Active Manager chose to mount the database copy after the best copy selection process, even after attempt copy last logs (ACLL) had run.

To help mitigate against the risk of data loss in a database failover such as this, Exchange Server 2013 has a feature called Safety Net, which stores copies of messages that have been successfully processed by a server for a configurable length of time (2 days by default).

In the event of a lossy failover, Active Manager is able to request the resubmission of missing email messages from Safety Net.

For more on Safety Net see the Transport High Availability chapter of this guide.

**Real World:** Safety Net also comes into play when activating a lagged database copy. When lagged copies exist in a DAG it is recommended to set the Safety Net hold time to a value equal to or greater than the replay lag time of the lagged copy.

## Circular Logging

As with previous versions of Exchange Server, Exchange Server 2013 uses transaction logging for its JET database engine. A single log stream per database is written to disk, consisting of multiple 1Mb transaction log files.

When a database backup occurs the transaction log files that recorded the transactions that had been committed to the database at the time of the backup are removed. This is often referred to as log file “truncation”. Most Exchange administrators will be familiar with the concept of running a database backup to “truncate” the log files and reclaim the disk space that they were consuming.

Circular logging for a database hosted on a standalone mailbox server, or for single copy databases, behaves slightly differently than standard transaction logging. Instead of waiting until a database backup is performed to truncate the log files, instead the log files are overwritten when the transactions have been committed to the database.

This has the effect of using less disk space on the server because the log files are not accumulating over the course of hours or days between backups being taken. However, it also means that for a database hosted on standalone mailbox servers that the transaction log files cannot be used to recover a database up to the point in time at which a failure occurred. Instead it can only be recovered to the point in time that the last backup was taken.

When a database is replicated between two or more DAG members circular logging behaves differently again. In this situation Continuous Replication Circular Logging (CRCL) is used instead of the standard (JET) circular logging used for standalone mailbox servers or single copy databases.

Exchange cannot switch seamlessly between the two modes of circular logging. When JET circular logging is enabled on a single database copy, a second copy of the database cannot be added until circular logging is disabled. Similarly, when CRCL is enabled on a database with two or more copies, the last passive copy cannot be removed (ie reverting the database to a single copy) until circular logging is disabled.

Because the transaction log files may still be required for replication they cannot simply be removed when a server has committed those transactions to its own copy of the database. Instead, when CRCL is enabled, some additional factors are taken into consideration first before a log file can be deleted.

Those factors are:

- Whether the transactions recorded in the log file have been committed to the database
- Whether the other non-lagged database copies agree that the log file can be deleted (ie, they have replicated and replayed the log file)
- Whether the lagged copies of the database have inspected their copy of the log file (ie, to verify it is not corrupted)
- For lagged copies, whether the log file is older than the replay lag time plus the truncation lag time
- For lagged copies, whether the log file has already been deleted from the server hosting the active database copy

**Real World:** Is CRCL safe to use? In some ways yes. The risk of data loss if a single database copy fails is mitigated by having multiple database copies in the DAG, which is why it is only recommended when there are at least 3 copies of each database in the DAG.

However there is still the risk of data loss if a data deletion or corruption occurs that replicates to all copies of the database, and you are unable to restore to a specific point in time to recover the lost data because CRCL is enabled. That risk can be mitigated by the use of lagged database copies.

## DAG Server and Storage Design

The server and storage design for an Exchange Server 2013 is critical to the success of your high availability deployment. In addition to performance considerations, some features such as auto-reseed are only possible with the correct design in place.

# Server Sizing Calculator

Any Exchange Server 2013 deployment should include the use of the server sizing calculator published by Microsoft. This calculator represents the best and latest sizing guidance available for Exchange Server 2013, based on real world deployments.

As such, the calculator does vary over time, with changes such as minor calculation fixes all the way up to major changes such as the additional CPU requirements for MAPI-over-HTTP released in Exchange Server 2013 Service Pack 1.

You can download the [Exchange 2013 Server Role Requirements calculator](#) from TechNet<sup>5</sup>. Always check to make sure you are downloading the latest version.

The screenshot shows the Exchange 2013 Server Role Requirements Calculator interface. It includes several configuration sections:

- Environment Configuration:**
  - Global Catalog Server Architecture: 64-bit
  - Server Multi-Role Configuration (MBX+CAS): Yes
  - Server Role Virtualization: No
  - High Availability Deployment: Yes
  - Number of Mailbox Servers Hosting Active Mailboxes / DAG (Primary Datacenter): 8
  - Number of Database Availability Groups: 8
- Site Resilience Configuration:**
  - Site Resilient Deployment: Yes
  - Site Resilience User Distribution Model: Active/Active (Single DAG)
  - Site Resilience Recovery Point Objective (Hours): 24
  - Activation Block Secondary Datacenter Mailbox Servers: No
- Mailbox Database Copy Configuration:**
  - Total Number of HA Database Copy Instances (Lagged Copy) within DAG: 3
  - Total Number of Lagged Database Copy Instances within DAG: 1
  - Number of HA Database Copy Instances Deployed in Secondary Datacenter: 1
  - Number of Lagged Database Copy Instances in Secondary Datacenter: 1
- Lagged Database Copy Configuration:**
  - Lagged Copy Log Replay Delay (Hours): 168
  - Lagged Copy Log Truncation Delay (Hours): 0
- Exchange Data Configuration:**
  - Data Overhead Factor: 0%
  - Mailbox Moves / Week Percentage: 1%
  - Dedicated Maintenance / Restore Volume?: Yes
  - Volume Free Space Percentage: 5%
  - Log Shipping Network Compression: Enabled
  - Log Shipping Compression Percentage: 30%
- Database Configuration:**
  - Maximum Database Size Configuration: Default
  - Minimum Database Size (GB): 100
  - Automatically Calculate Number of Unique Databases / DAG: Yes
  - Partition Number of Databases / DAG: 102
  - Calculate Number of Unique Databases / DAG for Symmetrical Distribution: Yes

A detailed walk through of the calculator is beyond the scope of this guide due to the evolving nature of the guidance it provides. However it is recommended that you refer to the following Microsoft content to better understand how the calculator is used:

- [Plan it the right way - Exchange Server 2013 sizing scenarios](#)<sup>6</sup> (MEC 2014 session recording)
- [Ask the Perf Guy: Sizing Exchange 2013 Deployments](#)<sup>7</sup>

## Multiple Databases per Volume

Multiple databases per volume is a new recommendation for Exchange Server 2013. It allows for both the database files and log files for multiple databases, including lagged copies, to be located on the same volume.

<sup>5</sup> <http://gallery.technet.microsoft.com/Exchange-2013-Server-Role-f8a61780>

<sup>6</sup> <http://channel9.msdn.com/Events/MEC/2014/ARC308>

<sup>7</sup> <http://blogs.technet.com/b/exchange/archive/2013/05/06/ask-the-perf-guy-sizing-exchange-2013-deployments.aspx>

This configuration makes better use of the capacity of newer hard disks, allows individual database sizes to be kept smaller rather than one very large database per volume, and can improve reseed times when a failed disk has been replaced.

For administrators used to placing each database on a dedicated volume, and the corresponding transaction log files on a separate volume, this may be a surprising recommendation to read.

However, consider that the investment Microsoft has made in reducing the IOPS requirements of databases removes the need to place each database on dedicated spindles. Microsoft also uses multiple databases per volume in Office 365, which has proven it to be a very efficient and cost effective way to manage database storage in a large scale environment.

Deciding on how many databases to place onto the same volume is simple – the number of databases per volume should equal the number of copies of each database. For example, if each database has three copies, then three database copies can be placed on each volume.

Multiple databases per volume can be deployed on either RAID or JBOD disks. That said, multiple databases per volume is only recommended when three or more database copies are configured, whether in the same or separate datacenters.

## RAID vs JBOD

As mentioned above, for a standalone Exchange 2013 server it makes sense to use a storage configuration that is resilient to single component failures, for example a RAID array.

However, when Exchange Server 2013 has been deployed in a database availability group, with multiple copies of each database across multiple servers the need for RAID storage diminishes.

In fact, with three or more database copies Microsoft recommends considering the use of JBOD (Just a Bunch of Disks) storage instead of RAID, as the database copies themselves provide the fault tolerance that a RAID array usually would. The benefit to you is that fewer physical disks are required, which is a cost saving, as well as removing the complexity of managing RAID configurations.

More specifically, the number of database copies in each datacenter and whether any lagged copies exist form the basis for Microsoft's recommendations for JBOD usage.

In a single datacenter deployment the recommendations are:

- RAID can be used for all volumes hosting Exchange databases
- JBOD can be used for volumes hosting Exchange databases provided that at least three copies of the database exist
- Lagged copies should be hosted on RAID unless two or more lagged copies exist

In a multiple datacenter deployment the recommendations are:

- RAID can be used for all volumes hosting Exchange databases
- JBOD can be used for volumes hosting Exchange databases provided that at least two copies of the database exist in the same datacenter
- Lagged copies can be hosted on JBOD as long as two lagged copies exist, whether they are both in the same or separate datacenters, or if a single lagged copy exists it must have log play down enabled.

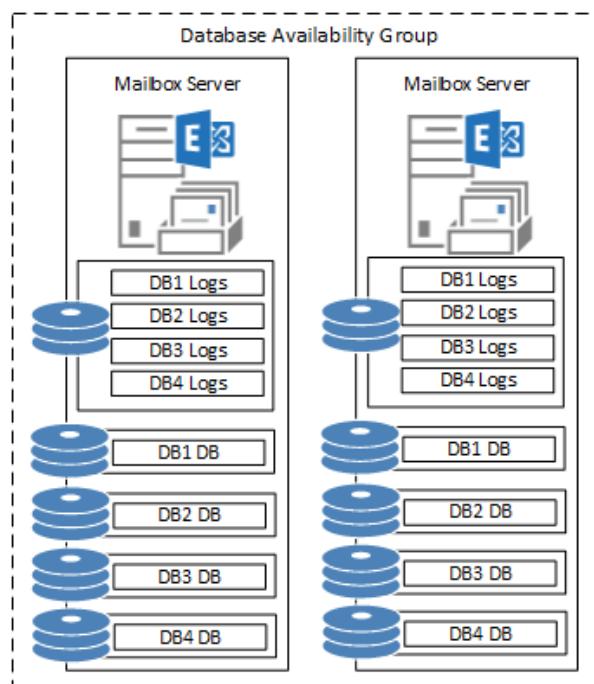
**Note:** The RAID vs JBOD decision consideration applies only to volumes that store Exchange databases and log files. It is still recommended to always use a RAID volume (usually RAID1) to host the operating system and Exchange program files, log files, and transport database.

## Disk Layout Examples

Considering the different storage options discussed in this section you can see that a variety of disk configurations are possible for Exchange Server 2013 DAGs.

### *Separate Log and Database Volumes*

What many administrators would be used to seeing is the traditional storage layout of dedicated log and database volumes, with those volumes backed by RAID storage.



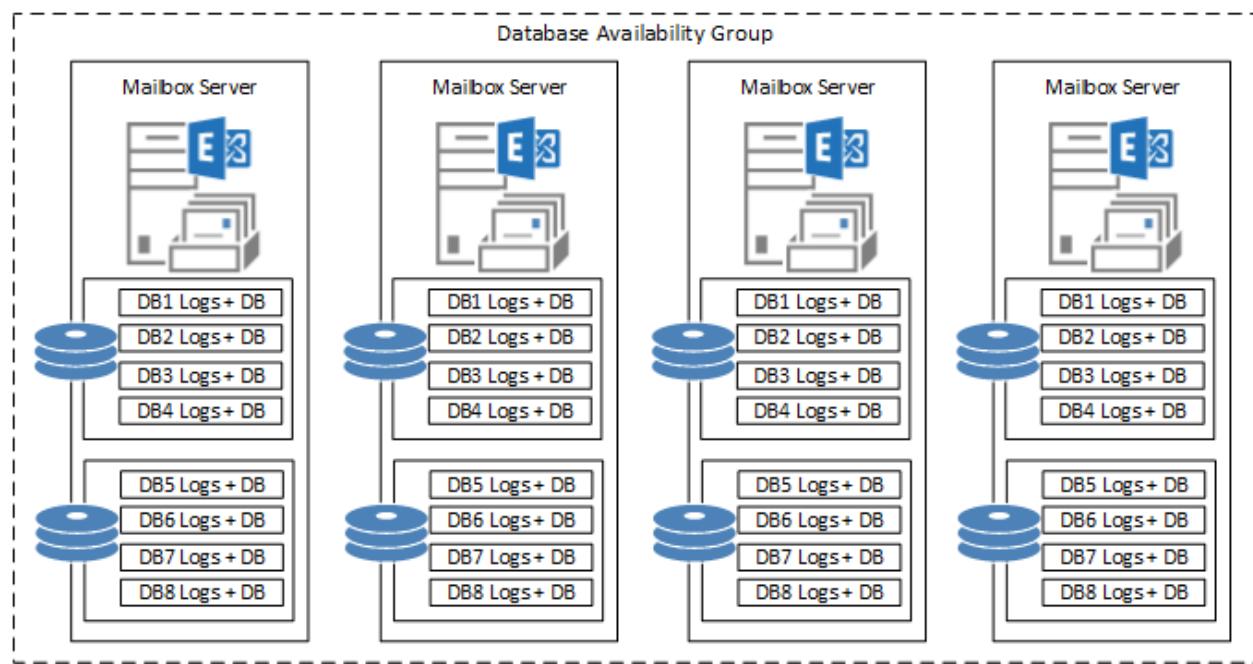
In Exchange Server 2013, even with a DAG deployed, this is still supported. In fact, with a smaller DAG deployment, such as two DAG members, it would still be recommended to use this type of configuration to protect from data loss due to disk failures.

Remember, multiple databases per volume is only recommended when three or more copies of each database exist.

#### *Multiple Databases per Volume*

When the recommended minimum three copies of databases exist then you may choose to consolidate the logs and database files for multiple databases onto single volumes.

For example, this four member DAG can host four database per volume on each of the DAG members.



You may choose RAID or JBOD storage however as discussed earlier this will depend on the number of datacenters and lagged copies involved. For example, with a single datacenter and only one lagged copy of each database it would not be recommended to place multiple databases per volume onto JBOD storage.

#### *Drive Letters vs Mount Points*

When we are configuring Exchange Server storage we have the choice to use individual drive letters for each volume, or to use mount points for each volume.

Use of drive letters is simple and familiar to just about every server administrator in the IT industry. However, as an Exchange server begins to scale up to host large numbers of mailboxes across more databases it can easily run out of available drive letters to assign to new volumes.

Using mount points instead of drive letters resolves this issue. Instead of a volume being assigned a drive letter it is mounted as a folder in an existing volume instead. Exchange Server 2013 fully supports the use of mount points and in many cases you will find it necessary to use them instead of drive letters.

#### *AutoReseed Considerations*

AutoReseed is a new feature of Exchange Server 2013 that can be used to quickly and automatically restore database redundancy after a storage failure has occurred.

In an AutoReseed configuration one or more spare disks are available for Exchange to use when an existing disk fails. The spare disk is mounted and the database copies that were hosted on the failed disk are automatically reseeded to the new disk.

Because this happens automatically it can mean that the only response required from support staff is to replace the failed disk and add it to the spare disk pool. All of the other recovery steps are performed automatically by Exchange Server 2013.

AutoReseed depends on the correct storage configuration being in place on the server. Specifically, AutoReseed is used in combination with JBOD, multiple databases per volume, and mount points instead of drive letters.

**Real World:** In the [“Preferred Architecture” for Exchange Server 2013](#)<sup>8</sup> Microsoft recommends JBOD disks with multiple databases per volume. Four database copies are configured per-disk, with activation preferences assigned such that under normal healthy conditions each disk only hosts one active database copy at a time. AutoReseed is also enabled.

Even if you do not immediately plan to use AutoReseed in your Exchange Server 2013 HA deployment it is recommended to use mount points instead of drive letters for your Exchange database and log volumes, even if you are using RAID volumes, unless you have a specific reason not to such as support for your monitoring or backup systems.

## Configuring Storage for Exchange Server 2013

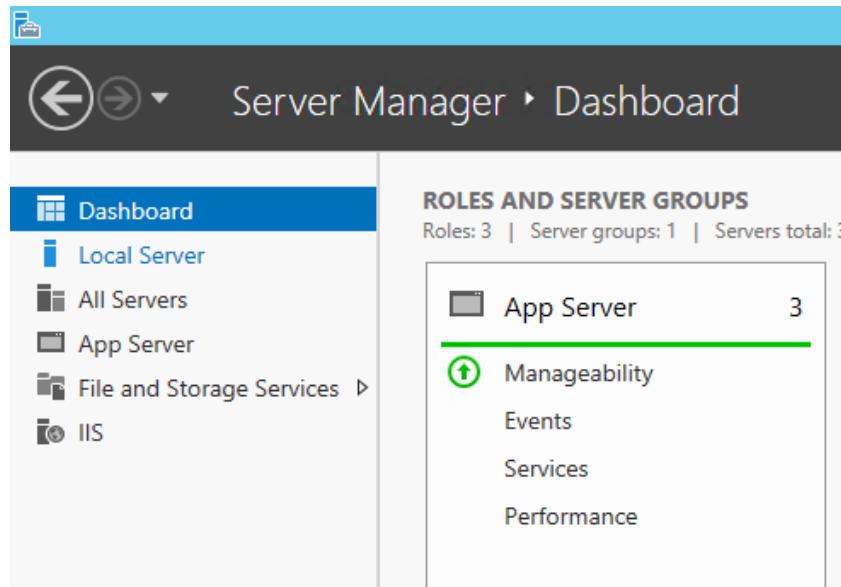
Let's take a closer look at configuring the disk layout for Exchange Server 2013. In our demo environment we'll assume that AutoReseed is going to be used, so we want to use mount points for the volumes instead of drive letters.

---

<sup>8</sup> <http://blogs.technet.com/b/exchange/archive/2014/04/21/the-preferred-architecture.aspx>

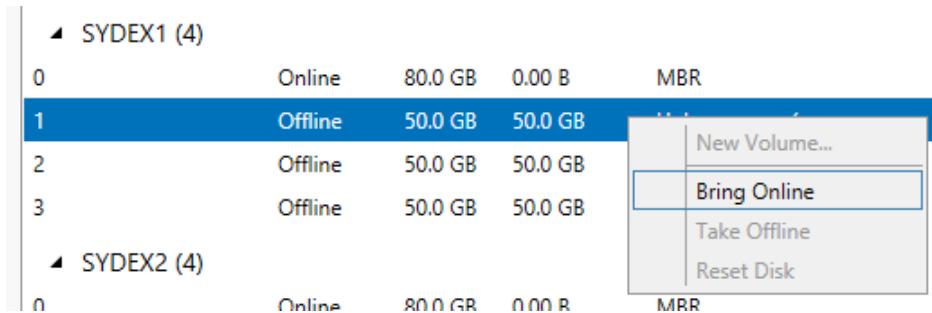
## Create New Volumes

In Server Manager select **File and Storage Services**.

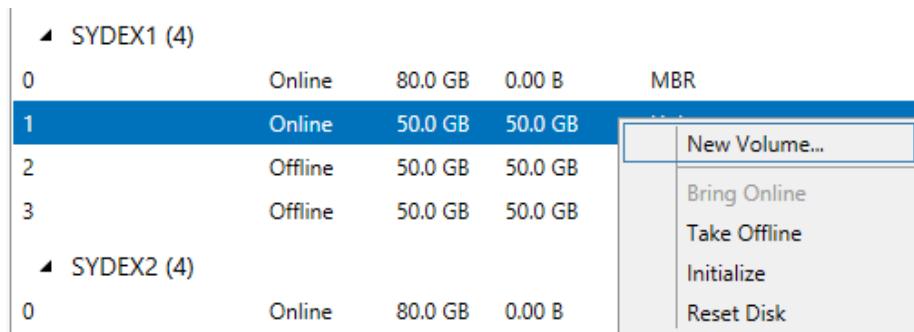


Select **Disks**, and then choose the first disk to configure, which should currently be offline.

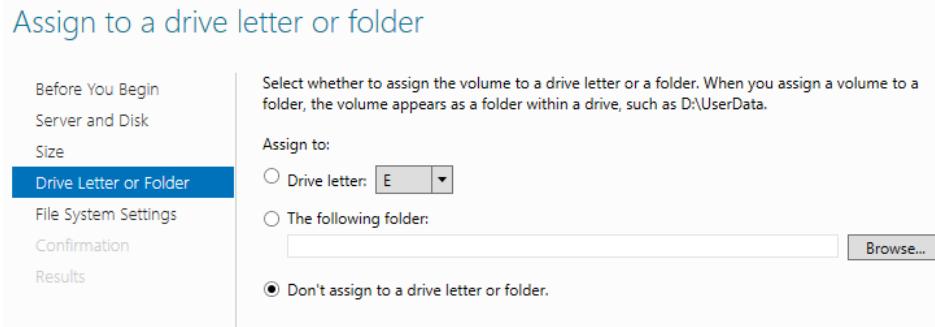
Right-click and choose **Bring Online**.



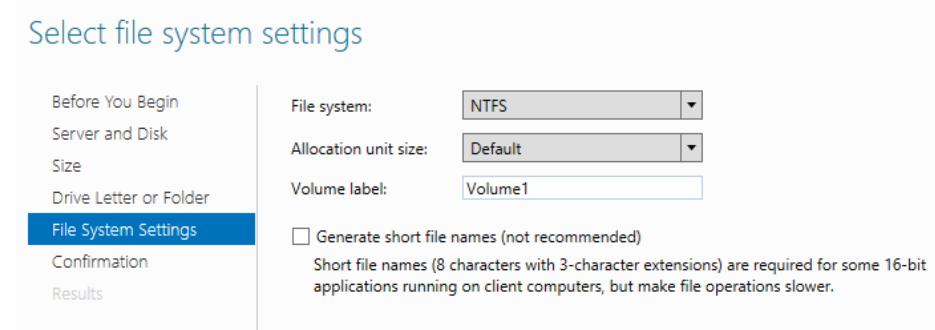
After the disk is online create a **New Volume**.



Step through the New Volume wizard. At the step for assigning a drive letter or folder choose **Don't assign to a drive letter or folder** for now.



Finally, give the volume a label and complete the New Volume wizard.



Repeat the process for each of the disks on both Exchange servers.

## Create Root Directories

The root directories that will hold the mount points for the Exchange storage.

When a database availability group is created it has two pre-configured values for these folder paths.

```
[PS] C:\>Get-DatabaseAvailabilityGroup | fl autodag*path
```

```
AutoDagDatabasesRootFolderPath : C:\ExchangeDatabases
AutoDagVolumesRootFolderPath   : C:\ExchangeVolumes
```

For simplicity it is recommended to use these default values.

Create the two folders using Windows Explorer or PowerShell.

```
[PS] C:\>New-Item -Path C:\ExchangeDatabases -Type Directory  
Directory: C:\  


| Mode  | LastWriteTime     | Length | Name              |
|-------|-------------------|--------|-------------------|
| d---- | 3/06/2014 8:39 PM |        | ExchangeDatabases |

  
[PS] C:\>New-Item -Path C:\ExchangeVolumes -Type Directory  
Directory: C:\  


| Mode  | LastWriteTime     | Length | Name            |
|-------|-------------------|--------|-----------------|
| d---- | 3/06/2014 8:39 PM |        | ExchangeVolumes |


```

## Configure the Number of Databases per Volume

As we discussed earlier, when using multiple databases per volume the number of databases should match the number of copies per database.

At this stage the database availability group has two members, which means each database can have two copies. Therefore, the number of databases per volume is 2.

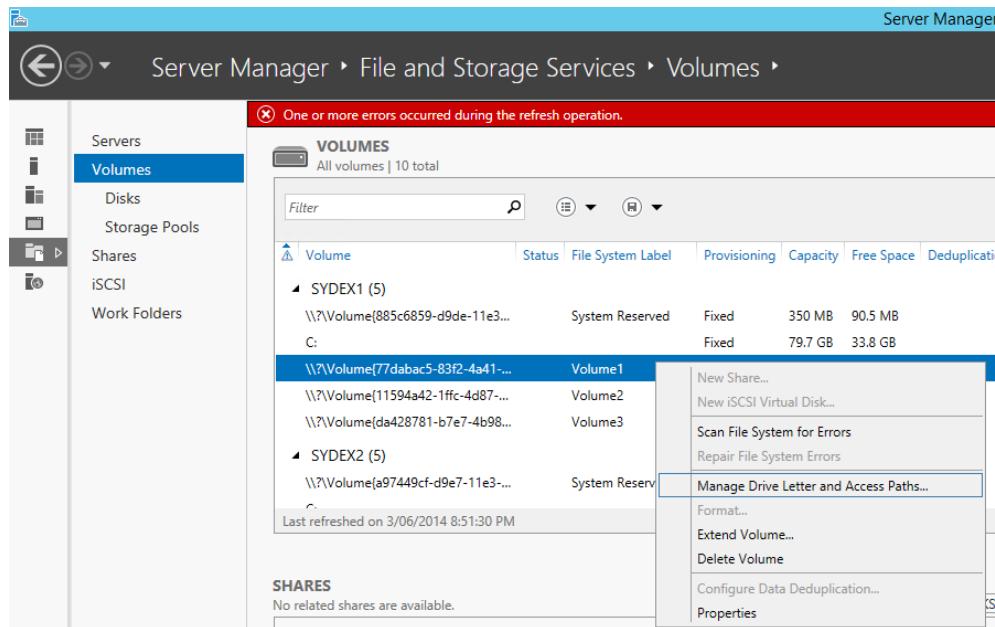
```
[PS] C:\>Set-DatabaseAvailabilityGroup DAG1 -AutoDagDatabaseCopiesPerVolume 2
```

**Note:** The AutoDagDatabaseCopiesPerVolume value is relevant for AutoReseed only. If you are not planning to configure AutoReseed then you can ignore this value. However, even without AutoReseed the use of mount points instead of drive letters, and multiple databases per volume, is still recommended.

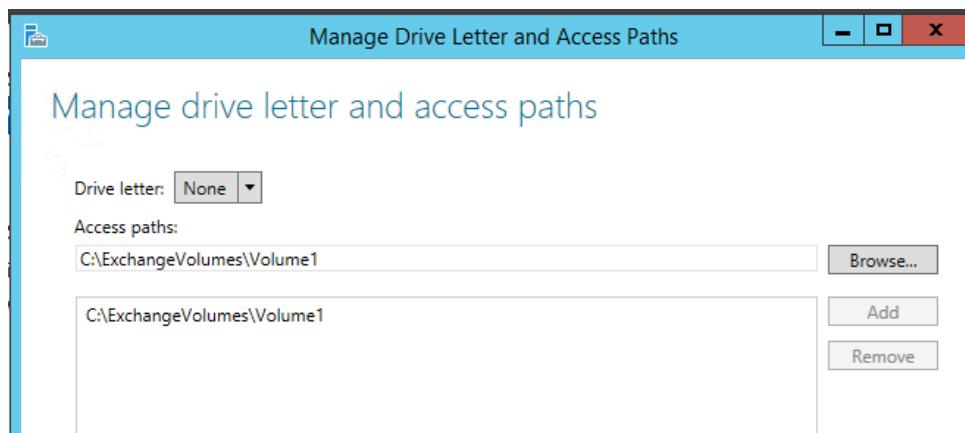
## Mount the Volume Folders

Each volume on the server is mounted into a separate volume folder under the root volume path. The folder names are not important however a simple approach is to use folder names that match the volume label you assigned earlier.

In Server Manager select **File and Storage Services**, then **Volumes**. Right-click the volume and choose **Manage Drive Letter and Access Paths**.



Enter the folder path and click **Add**. Then click **OK** to apply the change.



If the path you specified does not already exist you will be prompted to create it. Repeat this process for all of the volumes on each of the Exchange servers, mounting each one to a unique folder name in the root folder.

Volume	Status	File System Label	Provisioning	Capacity	Free Space
<b>SYDEX1 (5)</b>					
\?\Volume{885c6859-d9de-11e3...	System Reserved	Fixed	350 MB	90.5 MB	
C:		Fixed	79.7 GB	33.8 GB	
C:\ExchangeVolumes\Volume1	Volume1	Fixed	49.9 GB	49.8 GB	
C:\ExchangeVolumes\Volume2	Volume2	Fixed	49.9 GB	49.8 GB	
C:\ExchangeVolumes\Volume3	Volume3	Fixed	49.9 GB	49.8 GB	

## Create the Database Folders

Each database will require a unique folder path. These are configured as sub-folders of the databases root folder.

If you have used the server sizing calculator from Microsoft you should already know how many databases you need to configure. However for the purposes of this demonstration we will plan to configure a total of four databases.

Earlier we configured an AutoDagDatabaseCopiesPerVolume of 2, which means we can place two databases on each of the first two Exchange volumes available on the server, and then have one volume spare for AutoReseed, for a total of three volumes per DAG member.

**Note:** This scenario of two DAG members is used as an example only, before the DAG is expanded to four members later in this chapter. Remember, multiple databases per volume and JBOD configurations are only recommended when there are three or more database copies.

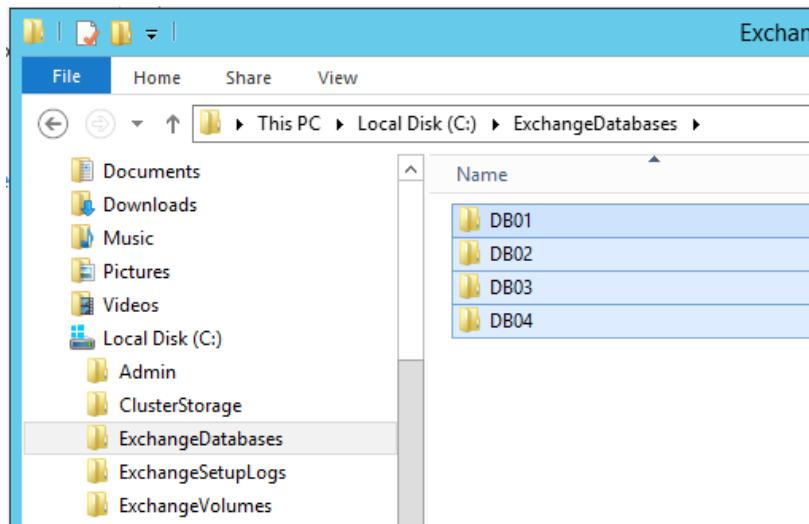
Using a database naming standard of “DBxx” where “xx” is the database number, our databases will be named:

- DB01
- DB02
- DB03
- DB04

The database folders must use the same name as the database itself. This means that our database folders will be as follows.

- C:\ExchangeDatabases\DB01
- C:\ExchangeDatabases\DB02
- C:\ExchangeDatabases\DB03
- C:\ExchangeDatabases\DB04

On each of the Exchange servers create the folders using Windows Explorer or PowerShell.



## Create the Mount Points for Databases

Each volume can now be mounted in the database folders that it will be hosting.

Because we are using multiple database per volume, with two copies per volume, each volume will be mounted in two folders.

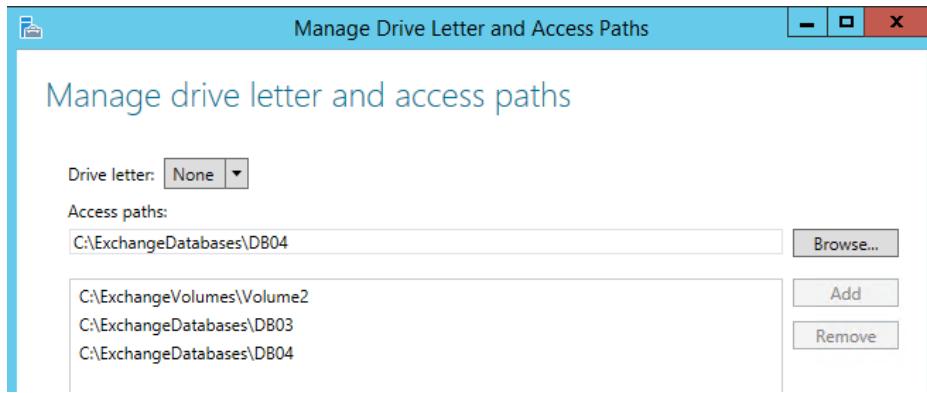
For example, Volume1 will be mounted in both C:\ExchangeDatabases\DB01 and C:\ExchangeDatabases\DB02.

This task can be performed using Server Manager, the Disk Management console, or the mountvol.exe command. For this demonstration Server Manager will be used.

On each Exchange server, right-click each of the two volumes that will be hosting databases and choose **Manage drive letter and access paths**. Add in the additional paths for each volume.



Repeat the process until each volume has been mounted to the database folders it will be hosting.



The third volume on each server is left as a spare for AutoReseed.

## Create the Database Directory Structure

Within each database folder a separate folder for the database files and the transaction log files is created. Again this must follow a naming convention that matches the database name. This means that in this demonstration the database directory structures will be as follows.

Database Name	Database Folder	Log Folder
DB01	C:\ExchangeDatabases\DB01\DB01.db	C:\ExchangeDatabases\DB01\DB01.log
DB02	C:\ExchangeDatabases\DB02\DB02.db	C:\ExchangeDatabases\DB02\DB02.log
DB03	C:\ExchangeDatabases\DB03\DB03.db	C:\ExchangeDatabases\DB03\DB03.log
DB04	C:\ExchangeDatabases\DB04\DB04.db	C:\ExchangeDatabases\DB04\DB04.log

On each Exchange server create the folders using Windows Explorer or PowerShell.

**Note:** You will notice that the same sub-folders such as DB01.db and DB01.log are visible in multiple database folders. This is normal, it is because the same underlying volume is mounted in both folders.

The Exchange storage is now configured and ready to host databases after Exchange Server 2013 has been installed and a database availability group has been created.

# DAG Network Design

A DAG network refers to a collection of one or more IP subnets that DAG members are directly connected to.

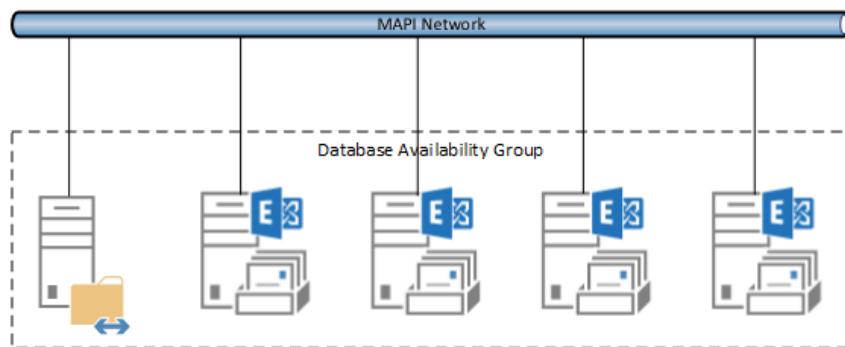
DAG networks must be configured correctly for healthy DAG operations. A new feature in Exchange Server 2013 is automatic configuration of DAG networks, which reduces the administrative effort required for correctly configuring DAG networks.

## DAG Network Types

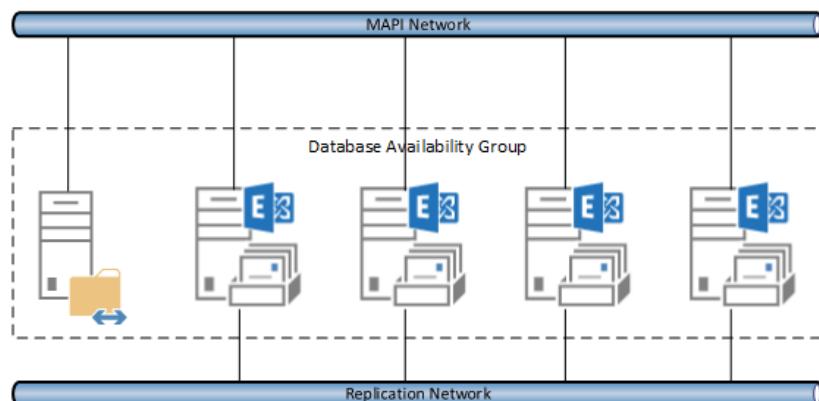
DAG networks are used for two purposes:

- Client traffic (sometimes referred to as the “MAPI network”)
- Database replication traffic (often referred to as “replication networks”)

Every DAG has at least one DAG network that is used for client and replication traffic.



A DAG can also have a number of additional networks dedicated to replication traffic.



## Dedicated Replication Networks

Dedicated replication networks can help reduce bandwidth utilization on the client-facing network which may improve network-related performance for client connections in very high volume environments.

A common configuration when dedicated replication networks exist is to configure the client-facing network to not participate in replication traffic. The replication traffic is then balanced across the replication networks using a simple “least recently used” (LRU) algorithm.

When this configuration is in place the DAG is smart enough to work out when all replication networks have failed and will resort to using the client-facing network so that replication can continue to occur.

## DAG Network Auto-Configuration

Exchange Server 2013 can automatically configure the available networks in a DAG to be client and replication networks as long as the network interfaces themselves are configured correctly.

The client-facing network interface on each DAG member should be configured with:

- A default gateway
- At least one DNS server
- The “Register this connection’s addresses in DNS” option enabled

The replication network interfaces, if there are any, should be configured with:

- No default gateway
- No DNS servers
- “Register this connection’s addresses in DNS” option disabled
- Static routes, if the network will span multiple IP subnets (e.g., in a multi-site DAG)

If these conditions are not met then DAG network automatic configuration may not produce the right results. However, in some cases automatic configuration doesn’t configure things quite right. In those situations we can simply disable automatic configuration and complete the DAG network configurations manually.

## Recommendations for DAG Networks

Although multiple DAG networks would appear to provide benefits for your Exchange Server 2013 deployment it may not necessarily be a good choice.

Multiple DAG networks creates an illusion of greater resilience when often there is no advantage at all because the DAG networks all share some common networking components, such as switches, rack cabling, or WAN connectivity.

If single points of failure can impact multiple DAG networks then all you have really achieved is adding complexity to your high availability deployment.

As such, it is recommended to not deploy multiple DAG networks unless there is a specific requirement identified, for example a risk of network interfaces being saturated by database replication or reseed traffic.

## Deploying and Managing a Database Availability Group

Now that we have covered the fundamentals of Exchange Server 2013 database availability groups let's start looking at the actual steps for deployment.

As we begin demonstrating deployment and management of Exchange 2013 DAGs we'll be using the demo environment outlined in Appendix A of this guide. Then, as we start to explore multi-site DAGs, we'll extend the environment with additional servers at a second site.

If you have the test lab resources to run the same number of servers in a multi-site configuration we certainly encourage you to do so.

### DAG Pre-Requisites

The pre-requisites for Exchange Server 2013 vary depending on the operating system you are installing on, and which Exchange server roles you are installing on the server.

The operating systems that are supported for Exchange Server 2013 are:

- Windows Server 2008 R2 with Service Pack 1 Enterprise edition
- Windows Server 2008 R2 RTM Datacenter edition
- Windows Server 2012 Standard or Datacenter edition
- Windows Server 2012 R2 Standard or Datacenter edition

In all cases a full server installation with GUI is required. Server Core installation is not supported for running Exchange Server 2013.

**Note:** Although Exchange Server 2013 can be installed on Windows Server 2008 R2 with Service Pack 1 Standard edition, the server would not be able to become a DAG member due to the lack of failover clustering features in Standard edition.

The operating system roles and features required for Exchange Server 2013 can change over time as updates are released. You can find the latest pre-requisites and associated PowerShell commands on TechNet:

- [Exchange 2013 Prerequisites](#)<sup>9</sup>

You can also refer Appendix A of this guide which walks through the process of installing Exchange 2013 pre-requisites on Windows Server 2012.

**Real World:** You may hear or read that the Microsoft Office Filter Pack is also a requirement for Exchange Server 2013. While this requirement was included in earlier guidance from Microsoft it is no longer the case. The capabilities provided by the Office Filter Pack for indexing certain file types in email attachments are now built in to the search features of Exchange 2013 itself. You can see a list of these file types by running Get-SearchDocumentFormat.

## Pre-Staging the Cluster Name Object

Database availability groups are built on an underlying failover cluster.

When installing a failover cluster a Cluster Name Object (CNO) is created. The Cluster Name Object (CNO) is simply a computer account in Active Directory that is used as the identity of the cluster.

When a DAG is created the failover cluster is not yet formed. It is only when the first Mailbox server is added as a DAG member that the failover cluster is created. This cluster creation process runs in the context of the local system of the Exchange 2013 server that is being added as a DAG member.

In environments where account creation is restricted, or when deploying Exchange 2013 on Windows Server 2012 or later, the Exchange server's local system credentials do not have rights to create computer accounts in Active Directory.

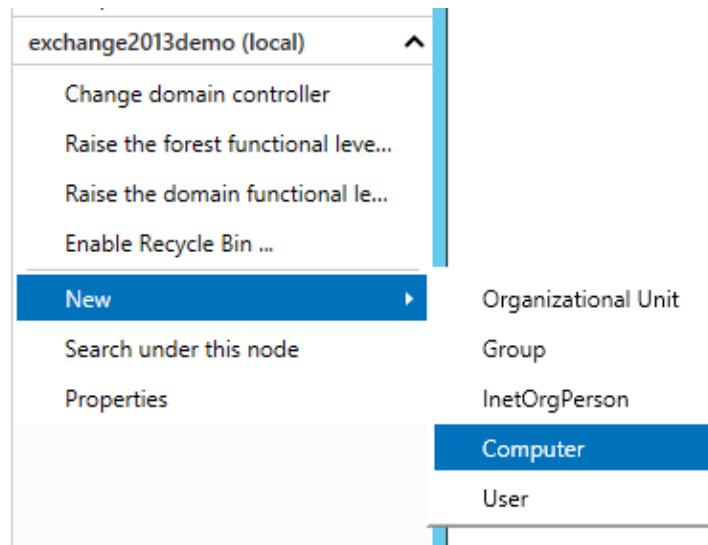
Therefore it is necessary to pre-stage the CNO in Active Directory.

In fact, even if you are deploying on Windows Server 2008 R2 where this requirement is not mandatory, it is still recommended to pre-stage the CNO.

**Note:** The exception to this is when you are deploying a DAG using Exchange Server 2013 SP1 or later, on Windows Server 2012 R2. In this scenario you have the *option* to deploy an IP-less DAG, also known as a "DAG without a cluster administrative access point (AAP)", which has neither a CNO nor does it have a DAG IP address.

<sup>9</sup> [http://technet.microsoft.com/en-us/library/bb691354\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/bb691354(v=exchg.150).aspx)

To pre-stage the CNO use the Active Directory Administrative Center to create a new computer account.

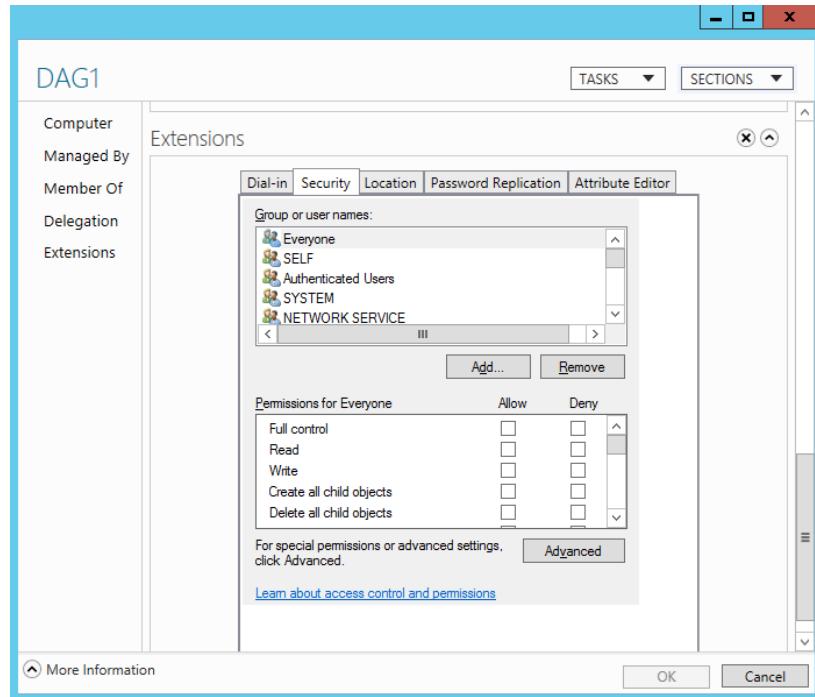


Give the computer account a meaningful name for your Exchange environment, such as DAG1. You can create the computer account in an OU that Exchange servers will be able to access, such as the default Computers container.

The dialog box has a header "Create Computer: DAG1" with "TASKS" and "SECTIONS" dropdowns. It has two main sections: "Computer" and "Managed By".

Computer	Computer
Managed By	(X) (A)
Member Of	
	Computer name: * DAG1
	Computer (NetBIOS) name: * DAG1
	Create in: CN=Computers,DC=exchange2013demo,DC=com <a href="#">Change...</a>
User or Group:	Default: Domain Admins <a href="#">Change ...</a>
	The above user or group can join this computer to a dom...
	<input type="checkbox"/> Assign this computer account as a Pre-Windows 2000 computer
	<input checked="" type="checkbox"/> Protect from accidental deletion

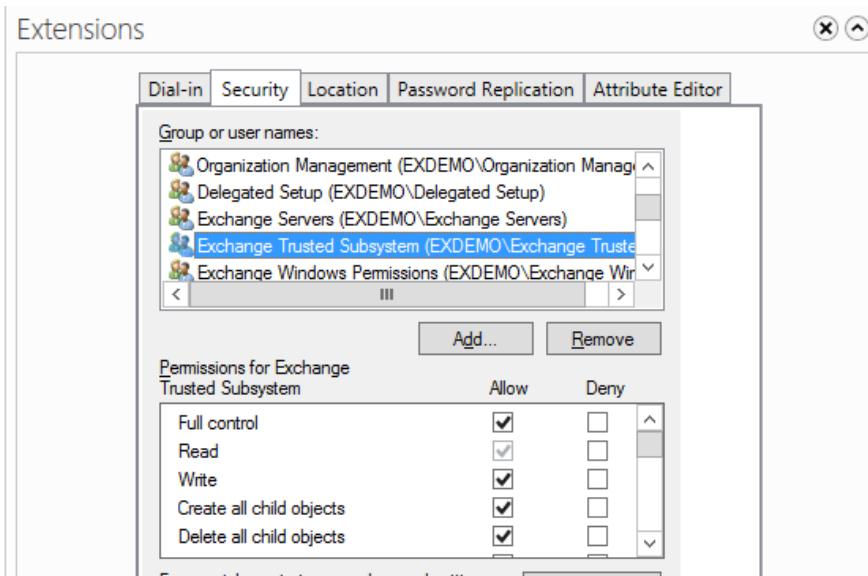
After creating the computer account open the Properties of the object, select Extensions and then the Security tab.



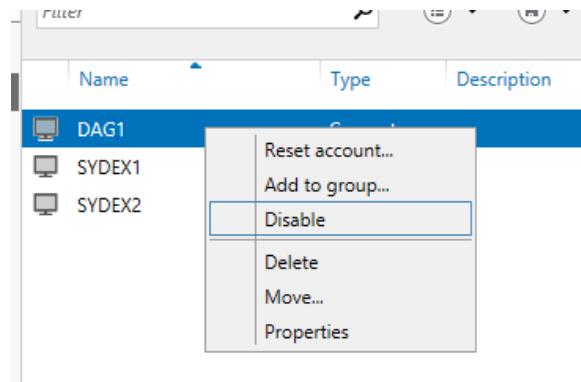
Click the Add button and add either:

- The computer account of the first Exchange server you intend to add as a DAG member (if you plan to add DAG members one at a time)
- The Exchange Trusted Subsystem group (if you plan to add all DAG members at once)

Grant Full Control permissions and click OK to apply the change.



Finally, disable the computer account.



**Real World:** DAGs without CNOs and IP addresses mean fewer potential issues with such problems as computer accounts being deleted or IP address conflicts occurring. At the scale of Office 365 this is obviously a big advantage.

In on-premises deployments those potential issues may be far less likely to occur. In addition to that, the requirement for a cluster administrative access point (AAP) may still exist for integration of third party applications such as backup software. Once the DAG has been deployed you cannot add/remove the AAP.

For these reasons, unless you are sure you do not need one, it is safer to deploy the DAG with an AAP.

## Preparing the File Share Witness

As with the Cluster Name Object the Exchange servers also need permissions assigned to the server that will be the file share witness for the DAG.

The file share witness server must be a member of the same Active Directory forest that the DAG is being created in. The following operating systems are supported for file share witness servers:

- Windows Server 2003 and 2003 R2
- Windows Server 2008 and 2008 R2
- Windows Server 2012 and 2012 R2

A single server can be the file share witness for multiple DAGs. While it may seem like a good idea to cluster the file share witness or make it somehow more highly available or resilient itself, keep in mind that any such configuration tends to add more complexity to your environment. Generally speaking the less complex your environment the easier it will be to manage.

When the file share witness is another Exchange server the correct permissions are already in place. When the file share witness is not another Exchange server, the permissions must be manually configured before the first Mailbox server is added as a DAG member.

For a member server in Active Directory the solution is to add the Exchange Trusted Subsystem group to the local Administrators group of the file share witness server.

Although it is not recommended to use a domain controller as the file share witness it is still supported to do so, and in fact we will use the domain controller of the demo environment as the file share witness for our DAG so that we don't need to add another server into the environment.

To use a domain controller as file share witness we simply add the Exchange Trusted Subsystem group into the Administrators group of the Active Directory domain where Exchange 2013 is being installed.

The screenshot shows the Windows Active Directory Users and Computers interface. On the left, there's a navigation pane with options like Group, Managed By, Member Of, Members, Password Settings, and Extensions. The main area is titled 'Administrators' and shows a list of group members under the 'Members' tab. A search bar labeled 'Filter' is at the top of the list. Below it, there are two entries: 'Enterprise Admins' and 'Exchange Trusted Subsystem'. The 'Exchange Trusted Subsystem' entry is highlighted with a blue selection bar at the bottom of the list. To the right of the list are 'Add...' and 'Remove' buttons. At the top right of the window, there are 'TASKS' and 'SECTIONS' dropdown menus.

**Warning:** The risks of this configuration should be obvious at this stage. By adding the Exchange Trusted Subsystem group to Administrators you are granting it full administrative privileges to the entire Active Directory domain. This means your Exchange server computer accounts, as well as anybody else who is deliberately or accidentally added to that group, will have those permissions in your domain.

## Creating a New Database Availability Group

An Exchange 2013 DAG can be created using either the Exchange Admin Center or the Exchange Management Shell.

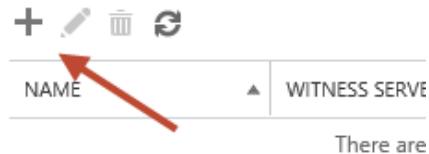
If this is your first time creating a DAG, or you are simply not comfortable with PowerShell yet, then using the Exchange Admin Center is fine.

## *Creating a Database Availability Group Using the Exchange Admin Center*

Log on to the Exchange Admin Center and navigate to servers → database availability groups.

The screenshot shows the Exchange Admin Center interface. The top navigation bar includes links for Enterprise, Office 365, Certificates, and database availability groups. The 'database availability groups' link is highlighted with a blue border. The main content area has a left sidebar with categories like recipients, permissions, compliance management, organization, protection, mail flow, mobile, public folders, unified messaging, and servers. The 'servers' category is also highlighted with a blue border. The main pane displays a table with columns for NAME, WITNESS SERVER, and MEMBER SERVERS. A message at the bottom of the table says, "There are no items to show in this view."

Create a new DAG.



Enter the name for the new DAG and the name of the witness server.

The screenshot shows the 'new database availability group' configuration form. It includes fields for the database availability group name (containing 'DAG1'), witness server ('syddc1.exchange2013demo.com'), and witness directory (an empty input field).

new database availability group	
*Database availability group name:	
DAG1	
Witness server:	
syddc1.exchange2013demo.com <input type="button" value="X"/>	
Witness directory:	
<input type="text"/>	

You can leave the witness directory blank and a default path on the C:\ volume of the server will be used. The witness directory does not consume significant disk space.

**Note:** If you don't specify a file share witness server then the DAG creation process will look for a Client Access server in the same Active Directory site that does not also have the Mailbox server role installed, and will choose that server to be the file share witness. Therefore, unless you have dedicated Client Access servers deployed, you will need to specify the name of the file share witness server.

For the DAG IP addresses you have three options for how you can proceed.

1. Enter a static IP address for each subnet that the DAG members are connected to for their MAPI network. This means a minimum of one IP address, but could mean several more if you have a multi-site DAG.
2. Leave the field blank and DHCP will be used to assign IP addresses to the DAG.
3. Enter an IP address of 255.255.255.255 if you are creating a DAG without an administrative access point.

For this demonstration a static IP address is assigned to the DAG.



**Warning:** Although you can change the DAG IP address later you cannot change from a static or dynamic IP to an IP-less configuration (a DAG without an AAP) using 255.255.255.255, and vice versa. If you do change your mind later, perhaps due to a third-party software support issue, then you must recreate the DAG from scratch.

### *Creating a Database Availability Group Using the Exchange Management Shell*

In the Exchange Management Shell the New-DatabaseAvailabilityGroup cmdlet is used to create a DAG.

To create the same DAG demonstrated above we can run the following command:

```
[PS] C:\>New-DatabaseAvailabilityGroup -Name DAG1 -WitnessServer syddc1.exchange2013demo.com -  
DatabaseAvailabilityGroupIpAddresses 192.168.0.103
```

Name	Member Servers	Operational Servers
DAG1	{}	

For an IP-less configuration, instead of using 255.255.255.255 as you would in the Exchange Admin Center, instead we would need to use ([System.Net.IPAddress]):None, for example:

```
[PS] C:\>New-DatabaseAvailabilityGroup -Name DAG1 -WitnessServer syddc1.exchange2013demo.com -  
DatabaseAvailabilityGroupIpAddresses ([System.Net.IPAddress]):None
```

## Managing Database Availability Group Members

After the DAG has been created we can add Mailbox servers as DAG members. This task can be performed using either the Exchange Admin Center or the Exchange Management Shell.

The Exchange 2013 pre-requisites do not require the Failover-Clustering role to be installed before adding a DAG member. This role is installed automatically on DAG members as you add them.

**Real World:** When adding new DAG members if you encounter an error “Some or All Identity References Could Not Be Translated” refer to [this article](#)<sup>10</sup> for the solution. The bug is also fixed in Cumulative Update 5.

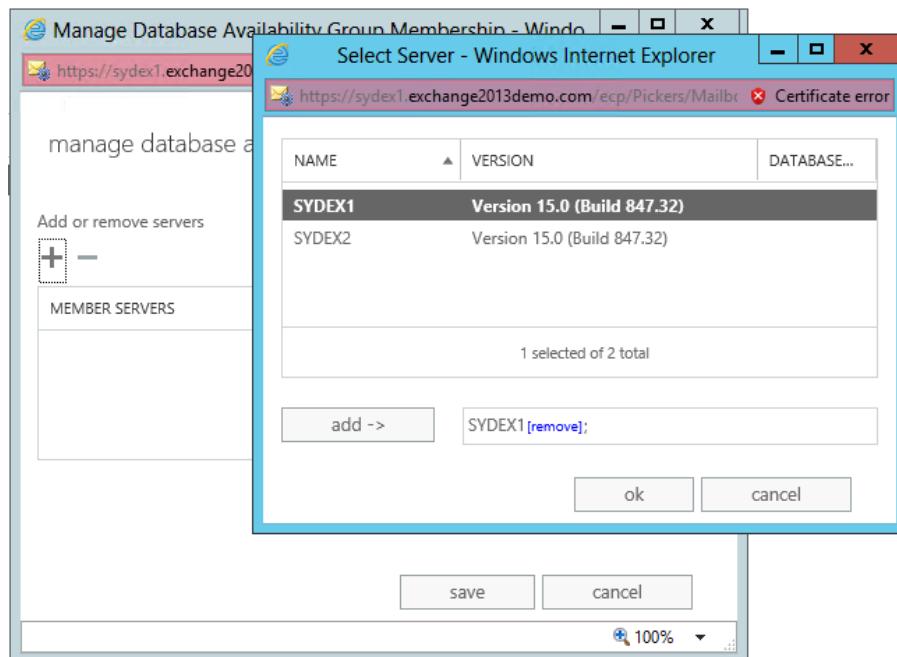
### *Adding a Database Availability Group Member Using the Exchange Admin Center*

Log on to the Exchange Admin Center and navigate to servers → database availability groups. Select the DAG you wish to add members to and click the “Manage DAG membership” button.

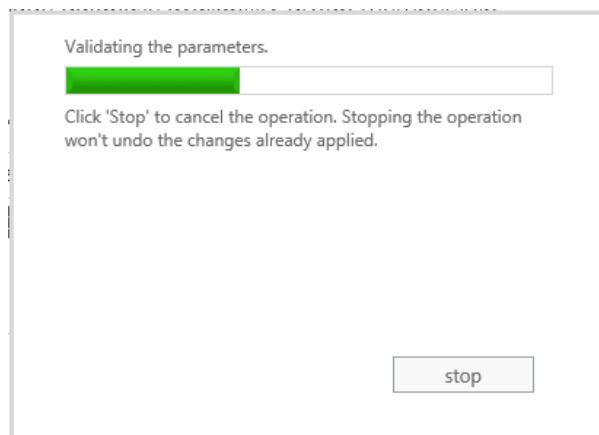
<sup>10</sup> <http://exchangeserverpro.com/error-identity-references-translated-exchange-2013-dag/>

NAME	WITNESS SERVER	MEMBER SERVERS
DAG1		syddc1.exchange20...

Click the Add button to add new DAG members. You can add multiple DAG members at the same time, however in this example a single DAG member is being added.



Click OK, then click Save, and wait for the change to process.



### *Adding a Database Availability Group Member Using the Exchange Management Shell*

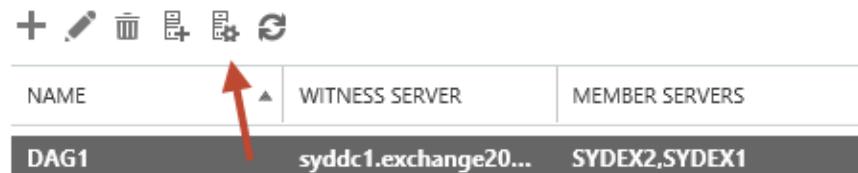
To add a DAG member with the Exchange Management Shell run the Add-DatabaseAvailabilityGroupServer cmdlet and provide the identity of the DAG and the name of the Mailbox server to be added.

```
[PS] C:\>Add-DatabaseAvailabilityGroupServer -Identity dag1 -MailboxServer sydex2
```

### *Removing a Database Availability Group Member Using the Exchange Admin Center*

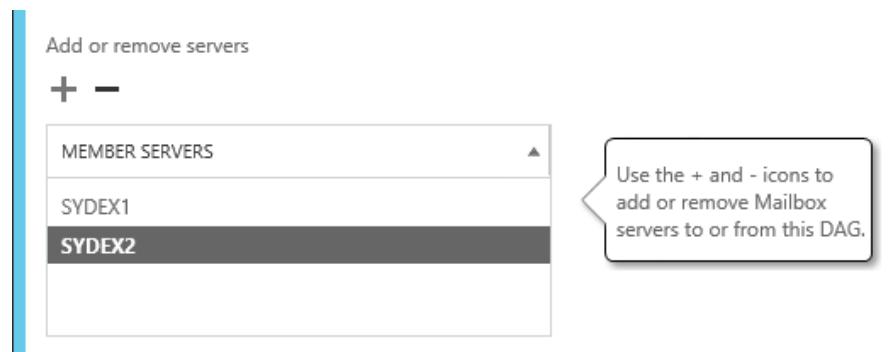
A DAG member can be removed from the DAG as long as it does not host an active database that has copies on other DAG members, or host copies of databases that are active on other DAG members. If you try to remove a DAG member when these conditions are not met you will receive an error and the process will fail.

Log on to the Exchange Admin Center and navigate to servers → database availability groups. Select the DAG you wish to remove members from and click the “Manage DAG membership” button.

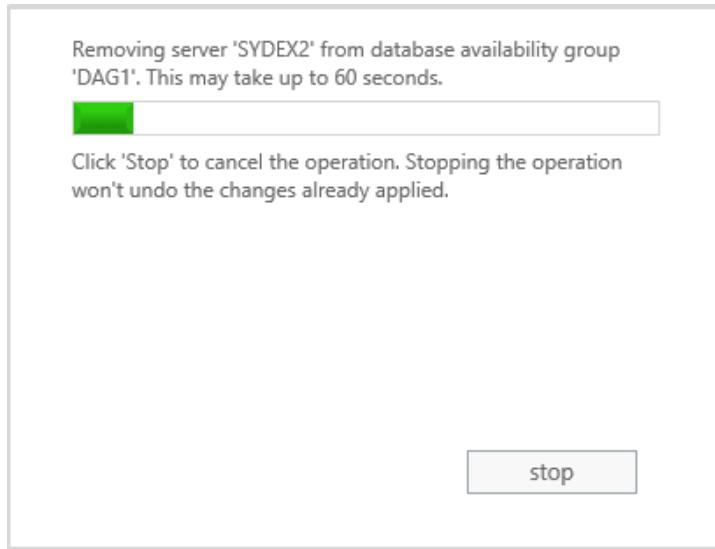


NAME	WITNESS SERVER	MEMBER SERVERS
DAG1	syddc1.exchange20...	SYDEX2,SYDEX1

Highlight the DAG members you wish to remove and then click the Remove button. You can remove multiple DAG members at the same time. If you remove all DAG members the underlying failover cluster is also removed.



Click Save and wait for the task to complete.



#### *Removing a Database Availability Group Member Using the Exchange Management Shell*

To remove a DAG member with the Exchange Management Shell run the Remove-DatabaseAvailabilityGroupServer cmdlet and provide the identity of the DAG and the name of the Mailbox server to be added.

```
[PS] C:\>Remove-DatabaseAvailabilityGroupServer -Identity dag1 -MailboxServer sydex2  
Confirm  
Are you sure you want to perform this action?  
Removing Mailbox server "SYDEX2" from database availability group "DAG1".  
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
```

## Managing DAG Networks

In Exchange Server 2013 the DAG network is configured automatically by default, with the ManualDagNetworkConfiguration attribute set to False.

```
[PS] C:\>Get-DatabaseAvailabilityGroup DAG1 | fl ManualDagNetworkConfiguration  
ManualDagNetworkConfiguration : False
```

With automatic DAG network configuration and a single network interface for each DAG member the DAG is able to automatically configure a single DAG network that is used for both MAPI and replication traffic.

```
[PS] C:\>Get-DatabaseAvailabilityGroupNetwork | fl

RunspaceId      : 0378f5aa-3c33-4660-a25e-82d442fe14b1
Name            : MapiDagNetwork
Description     :
Subnets         : {{192.168.0.0/24,Up}, {192.168.1.0/24,Up}}
Interfaces      : {{MELEX1,Up,192.168.1.101}, {MELEX2,Up,192.168.1.102},
                  {SYDEX1,Up,192.168.0.101}, {SYDEX2,Up,192.168.0.102}}
MapiAccessEnabled : True
ReplicationEnabled : True
IgnoreNetwork   : False
Identity        : DAG1\{MapiDagNetwork
IsValid         : True
ObjectState     : New
```

If multiple network interfaces exist on each DAG member and the network interface configurations are not correct, then DAG network auto-configuration will fail.

This is indicated by the subnets marked as “misconfigured”.

```
[PS] C:\>Get-DatabaseAvailabilityGroupNetwork | select Name,Subnets,Interfaces | fl

Name      : MapiDagNetwork
Subnets   : {{192.168.0.0/24,Misconfigured}, {192.168.1.0/24,Misconfigured},
           {10.1.100.0/24,Misconfigured}}
Interfaces : {{MELEX1,Up,192.168.1.101}, {MELEX2,Up,192.168.1.102},
             {SYDEX1,Up,192.168.0.101}, {SYDEX1,Up,10.1.100.2},
             {SYDEX2,Up,192.168.0.102}, {SYDEX2,Up,10.1.100.3}}

Name      : ReplicationDagNetwork01
Subnets   : {{10.1.101.0/24,Up}}
Interfaces : {{MELEX1,Up,10.1.101.2}, {MELEX2,Up,10.1.101.3}}
```

Although the best practice recommendation is to use a single DAG network, if for some reason you have determined that multiple networks are required in your environment then the first steps to resolve misconfigured DAG networks is to correct the network interface configurations to align with the requirements listed earlier in this chapter.

Next, enable and then re-disable manual DAG network configuration.

```
[PS] C:\> Get-DatabaseAvailabilityGroup DAG1 -ManualDagNetworkConfiguration $true
[PS] C:\> Get-DatabaseAvailabilityGroup DAG1 -ManualDagNetworkConfiguration $false
```

When you query the DAG networks again you should no longer see misconfigured subnets if you have configured your network adapters correctly.

```
[PS] C:\>Get-DatabaseAvailabilityGroupNetwork | select Name,Subnets,Interfaces | fl

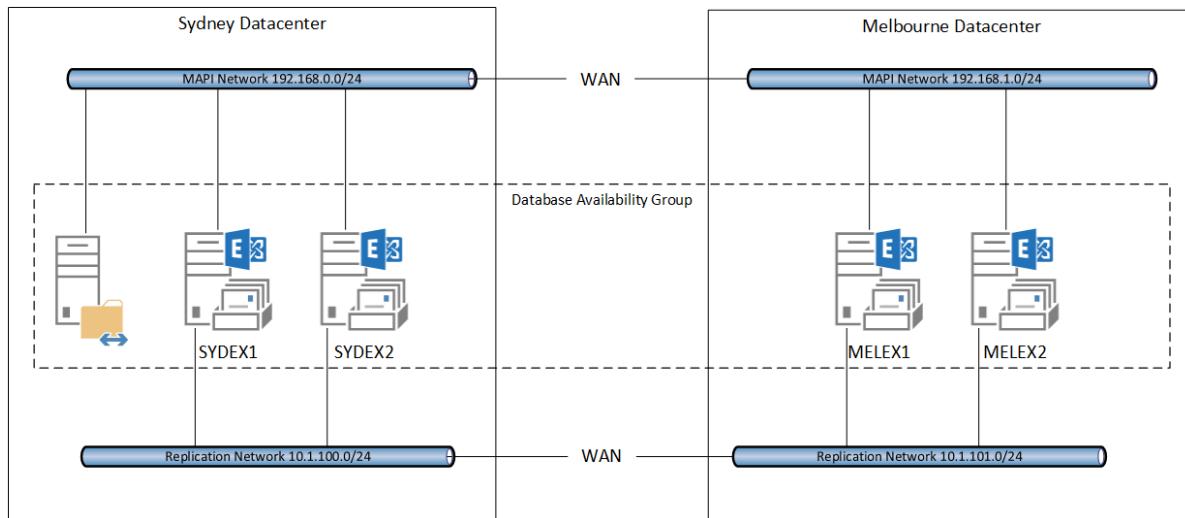
Name      : MapiDagNetwork
Subnets   : {{192.168.0.0/24,Up}, {192.168.1.0/24,Up}}
Interfaces : {{MELEX1,Up,192.168.1.101}, {MELEX2,Up,192.168.1.102}, {SYDEX1,Up,192.168.0.101}, {SYDEX2,Up,192.168.0.102} }

Name      : ReplicationDagNetwork01
Subnets   : {{10.1.101.0/24,Up}}
Interfaces : {{MELEX1,Up,10.1.101.2}, {MELEX2,Up,10.1.101.3} }

Name      : ReplicationDagNetwork02
Subnets   : {{10.1.100.0/24,Up}}
Interfaces : {{SYDEX1,Up,10.1.100.2}, {SYDEX2,Up,10.1.100.3} }
```

The correct network interface configuration results in no misconfigured subnets. However in the example above the DAG network auto-configuration has created two replication networks instead of the expected one

This happens because the DAG members are located in multiple sites and the DAG member network interfaces connected to those subnets (10.1.100.0/24 and 10.1.101.0/24) have no default gateway configured, so they can't communicate with each other even though a physical connection exists.



In this situation static routes can be added to the servers so that the replication network interfaces on the two subnets are able to communicate.

To resolve this we can add static routes to each DAG member so that the replication interfaces can communicate. For example, on SYDEX1 and SYDEX2 the following route is added.

```
PS C:\> New-NetRoute -DestinationPrefix "10.1.101.0/24" -InterfaceAlias Replication -NextHop 10.1.100.254
```

And on MELEX1 and MELEX2 the following route is added.

```
PS C:\> New-NetRoute -DestinationPrefix 10.1.100.0/24 -InterfaceAlias Replication -NextHop 10.1.101.254
```

By disabling and then re-enabling DAG auto-configuration we can force the DAG to re-assess the network configuration, and this time we get the desired outcome of two DAG networks.

```
[PS] C:\>Set-DatabaseAvailabilityGroup dag1 -ManualDagNetworkConfiguration:$true  
[PS] C:\>Set-DatabaseAvailabilityGroup dag1 -ManualDagNetworkConfiguration:$false  
[PS] C:\>Get-DatabaseAvailabilityGroupNetwork | select Name,Subnets,Interfaces | fl  
  
Name      : MapiDagNetwork  
Subnets   : {{192.168.0.0/24,Up}, {192.168.1.0/24,Up}}  
Interfaces : {{MELEX1,Up,192.168.1.101}, {MELEX2,Up,192.168.1.102}, {SYDEX1,Up,192.168.0.101},  
             {SYDEX2,Up,192.168.0.102}}  
  
Name      : ReplicationDagNetwork01  
Subnets   : {{10.1.101.0/24,Up}, {10.1.100.0/24,Up}}  
Interfaces : {{MELEX1,Up,10.1.101.2}, {MELEX2,Up,10.1.101.3}, {SYDEX1,Up,10.1.100.2},  
             {SYDEX2,Up,10.1.100.3}}4
```

Finally, if there were other network interfaces in the servers that should not be used by the DAG at all, such as iSCSI interfaces or remote management ports, we can configure them to be ignored by the DAG so that it does not attempt to use them for replication traffic.

```
[PS] C:\>Set-DatabaseAvailabilityGroupNetwork DAG1\ReplicationDagNetwork02 -Name "iSCI" -  
ReplicationEnabled:$false -IgnoreNetwork:$true
```

## Managing Database Copies

Within an Exchange Server 2013 DAG copies of mailbox databases can exist on as many as 16 DAG members. Administrators perform a number of management tasks for database copies, including adding, removing, suspending, resuming, reseeding, and removing.

### *Moving Existing Databases before Adding Copies*

When the Exchange 2013 Mailbox server role is installed a database is created on the server by default with a randomly generated unique database name.

For example, on SYDEX1 the following database was created during setup.

```
[PS] C:\>Get-MailboxDatabase -Server SYDEX1

Name          Server      Recovery     ReplicationType
----          -----      -----       -----
Mailbox Database 1055054279  SYDEX1    False        None
```

The database was also placed in a default path, located within the Exchange Server 2013 program files directory.

```
[PS] C:\>Get-MailboxDatabase -Server SYDEX1 | fl EdbFilePath,LogFolderPath

EdbFilePath   : C:\Program Files\Microsoft\Exchange Server\V15\Mailbox\Mailbox Database
                  1055054279\Mailbox Database 1055054279.edb
LogFolderPath : C:\Program Files\Microsoft\Exchange Server\V15\Mailbox\Mailbox Database 1055054279
```

To align with the database naming requirements of AutoReseed and the storage layout that was created earlier in this chapter, we can rename this database and move the database and log files to the desired paths.

The Move-DatabasePath cmdlet will automatically dismount the database, move the required files to the new paths, updating the configuration data in Active Directory, and then remount the database again.

```
[PS] C:\>Set-MailboxDatabase "Mailbox Database 1055054279" -Name "DB01"

[PS] C:\>Move-DatabasePath -Identity "DB01" -EdbFilePath C:\ExchangeDatabases\DB01\db01.db\DB01.edb
      -LogFolderPath C:\ExchangeDatabases\DB01\db01.log

Confirm
Are you sure you want to perform this action?
Moving database path "DB01".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y

Confirm
To perform the move operation, database "DB01" must be temporarily dismounted, which will make it
inaccessible to all users. Do you want to continue?
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
```

We can repeat the same process for the default database on SYDEX2, renaming it to DB02 and moving it to the correct folder paths.

**Note:** The database move is easier to perform when the database has only one copy. After one or more database copies are added the Move-DatabasePath cmdlet is not able to automatically move the files for you, and instead you must perform several manual steps. When multiple copies of a database exist it is better to create a new database in the new location, and then perform mailbox moves. This avoids downtime for your end users.

#### *Creating New Mailbox Databases for a Database Availability Group*

So far we've renamed and moved the two default databases on SYDEX1 and SYDEX2 to become the DB01 and DB02 databases on the servers. Now we can create two additional database to complete the four database design that is being used in this example.

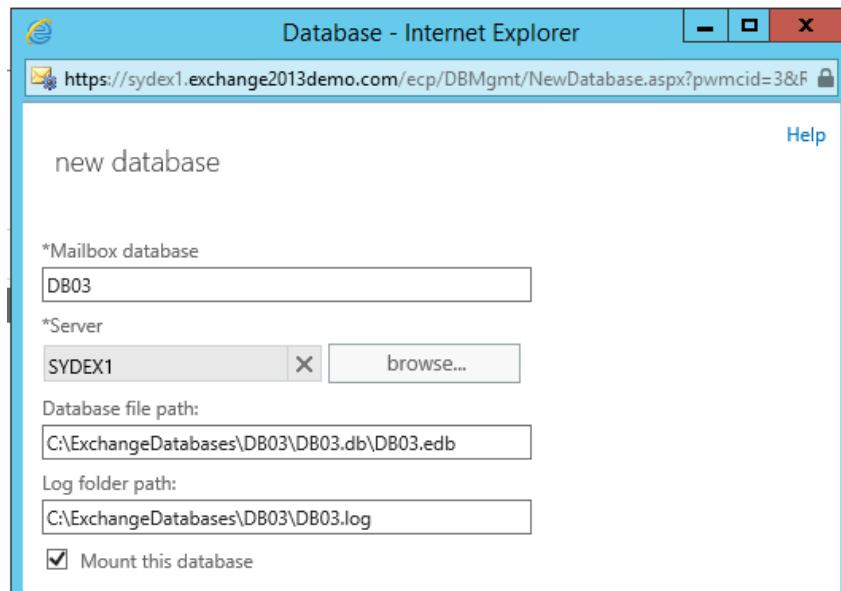
Mailbox databases can be created using the Exchange Admin Center. Navigate to **Servers → Databases** and click the button to create a new database.

The screenshot shows the Exchange Admin Center interface. At the top, there's a blue header bar with the Microsoft logo, 'Enterprise', and 'Office 365'. Below the header, the title 'Exchange admin center' is displayed. On the left side, there's a vertical navigation menu with links: 'recipients', 'servers', 'databases' (which is highlighted in blue), 'database availability groups', 'virtual directorie', 'permissions', 'compliance management', 'organization', 'protection', and 'mail flow'. In the center, there's a table titled 'Databases' with the following data:

NAME	ACTIVE ON SER...	SERVERS WITH COPIES	STAT...	BAD COPY COU...
DB01	SYDEX1	SYDEX1	Mo... 0	
DB02	SYDEX2	SYDEX2	Dism... 0	

Enter a name for the new database and choose a server to create it on. All databases are first created on a single server even when they are planned to be replicated between multiple servers in a database availability group.

Enter the database file path and log folder path following the required naming standard for AutoReseed.



Click **Save** to create and mount the database.

We can also use PowerShell to create the new database.

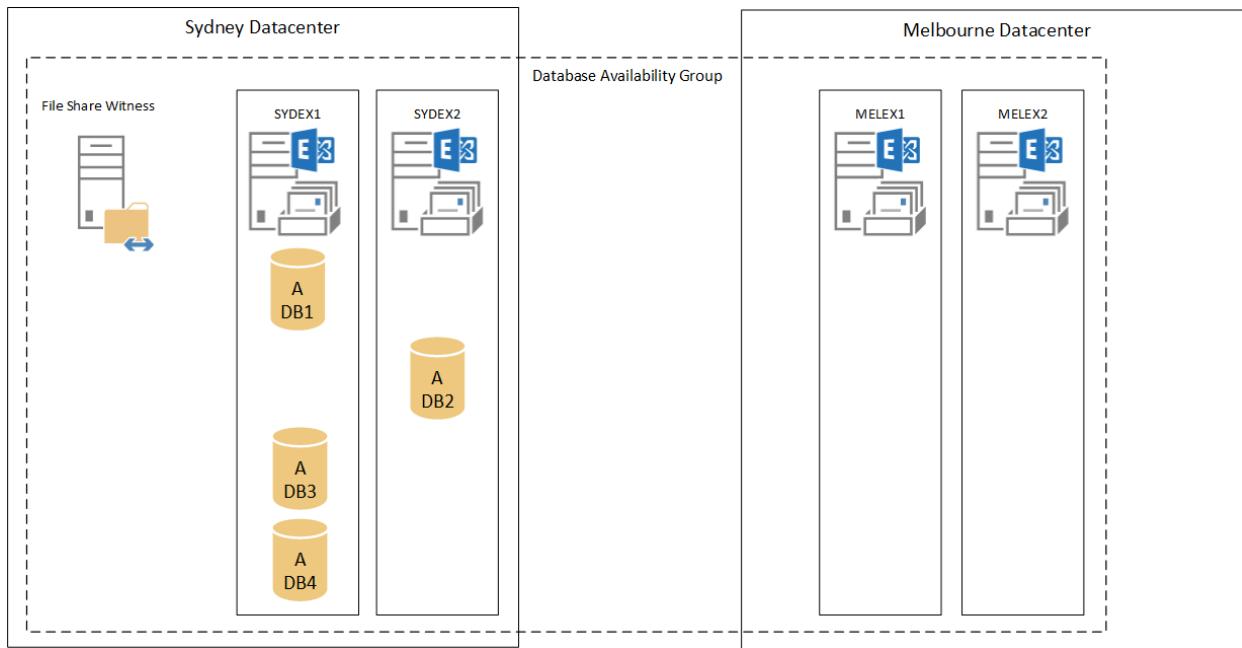
```
[PS] C:\>New-MailboxDatabase -Name DB04 -Server SYDEX1 -EdbFilePath  
C:\ExchangeDatabases\DB04\DB04.db\DB04.edb -LogFolderPath C:\ExchangeDatabases\DB04\DB04.log
```

Name	Server	Recovery	ReplicationType
DB04	SYDEX1	False	None

**Real World:** Often the first mount attempt for a new database fails due to Active Directory replication delays. If this occurs in your environment simply wait a few minutes before trying again.

Repeat the process above to create any additional databases that are planned for your environment.

In this example scenario four total databases are created on Mailbox servers in the DAG.



However, none of the databases have been configured for replication to other DAG members. We'll take a look at how to add database copies in the next section.

**Note:** When you create a new database in Exchange Server 2013 a warning will appear that instructs you to restart the Microsoft Exchange Information Store service. For more information on why this occurs refer to the article [Warning: Please Restart the Microsoft Exchange Information Store Service After Adding New Mailbox Databases<sup>11</sup>](#).

#### *Adding a Database Copy*

Each database in a DAG can have up to 16 total copies. A database begins with a single copy and additional copies need to be added by an administrator.

You can add one copy to a database at a time, as you will need to wait for each new copy to finish the seeding process before you can add the next copy for that database. However, you can add copies to multiple databases at the same time. Just be aware that the seeding traffic that this generates may impact your network performance, especially if you are seeding multiple databases across a WAN.

Database copies can be added using the Exchange Admin Center. Navigate to Servers → Databases and select a database.

---

<sup>11</sup> <http://exchangeserverpro.com/exchange-2013-please-restart-exchange-information-store-service-after-adding-new-databases/>

Click the button to reveal the menu option to add a database copy.

The screenshot shows the Exchange admin center interface. On the left, there's a navigation pane with links like recipients, permissions, compliance management, organization, protection, mail flow, and mobile. The main area is titled "Exchange admin center" and has a breadcrumb trail: servers > databases. Below the title is a toolbar with icons for add (+), edit (pencil), delete (trash), search (magnifying glass), refresh (refresh), and more options (ellipsis). To the right is a table titled "databases" with columns: NAME, AC, COPIES, and STATUS. The table contains five rows:

NAME	AC	COPIES	STATUS
DB01	SY	Add database copy Dismount	Mounted
DB02	SYDEX2	SYDEX2	Mounted
DB03	SYDEX1	SYDEX1	Mounted
DB04	SYDEX1	SYDEX1	Mounted

Choose the DAG member to add a copy of the database to. The activation preference will automatically choose the next available number. There are also additional options such as setting a replay lag time and postponing the seeding operation.

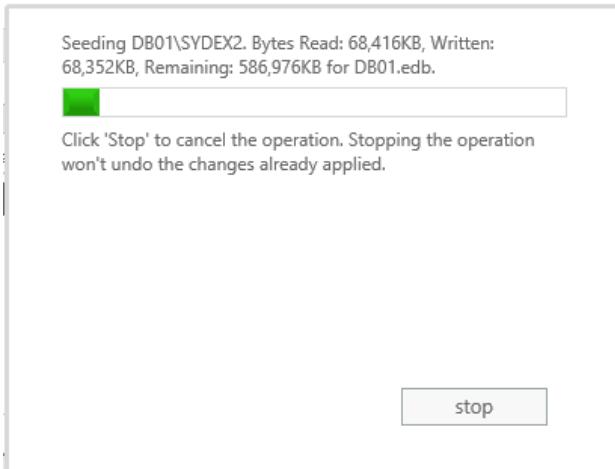
For this example scenario we are simply adding another database copy for DB01 to the server SYDEX2.

The screenshot shows the "Add Mailbox Database Copy - Internet Explorer" dialog box. It has the following fields:

- Mailbox database name: DB01
- \*Specify Mailbox server: SYDEX2
- Activation preference number: 2
- Servers hosting a copy of this database: SYDEX1
- Replay lag time (days): (empty input field)
- Postpone seeding

A callout bubble points to the "Specify Mailbox server" field with the text: "Use this field to select the DAG member on which you want to add the new database copy. Click browse..., select the server that will host the copy from list, and click OK."

Click **Save** to complete the change. The database copy will be configured and then the seeding process will run to copy the database and log files to the server.



We can also add database copies using PowerShell.

```
[PS] C:\>Add-MailboxDatabaseCopy -Identity DB02 -MailboxServer SYDEX1
```

### *Suspending and Resuming Database Copies*

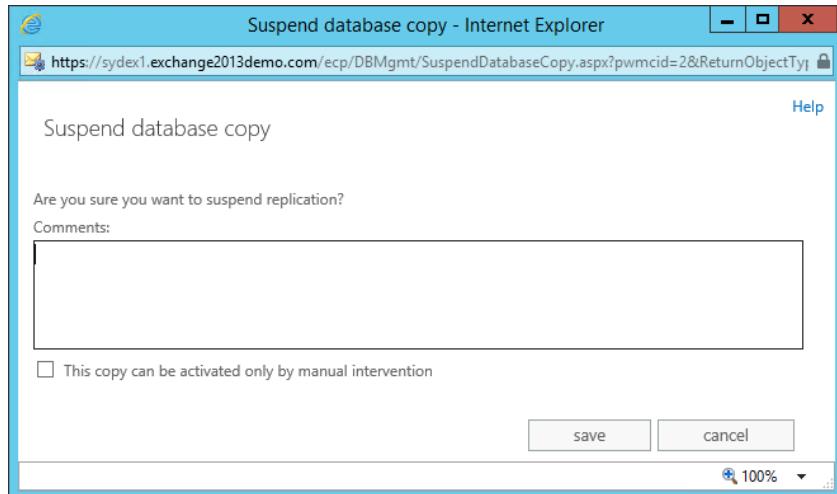
The continuous replication process can be suspended for a database copy. This may be necessary for reasons such as performing server maintenance, or for troubleshooting a problem.

To suspend a database copy we can use the Exchange Admin Center. After selecting a database click on **Suspend** in the right-pane of the webpage.

NAME	ACTIVE ON SER...	SERVERS WITH COPIES	STATUS	BAD COPY...
DB01	SYDEX1	SYDEX1,SYDEX2	Mounted	0
DB02	SYDEX2	SYDEX2	Mounted	0
DB03	SYDEX1	SYDEX1	Mounted	0
DB04	SYDEX1	SYDEX1	Mounted	0

DB01  
Database availability group:  
DAG1  
  
Servers  
SYDEX1  
SYDEX2  
  
Database copies  
DB01\SYDEX1  
Active Mounted  
Copy queue length: 0  
Content index state: Healthy  
[View details](#)  
DB01\SYDEX2  
Passive Resynchronizing  
Copy queue length: 2068  
Content index state: Healthy  
[Suspend | Activate | Remove](#)  
[View details](#)

You can add comments that explain the reason for suspending the database copy, and there is also an option to block automatic activation of the suspended database copy.



The database copy can also be suspended using PowerShell. When we reference a database copy in PowerShell the naming syntax is <Database Name>\<Server Name>, for example DB01\SYDEX2.

```
[PS] C:\>Suspend-MailboxDatabaseCopy DB01\SYDEX2
```

**Note:** You can't suspend the active database copy. Only a passive database copy can be suspended from replication.

Database copies can be resumed in much the same way. In the Exchange Admin Center click on **Resume** for the database copy.

Database copies

DB01\SYDEX1  
Active Mounted  
Copy queue length: 0  
Content index state: Healthy

[View details](#)

DB01\SYDEX2  
Passive Suspended  
Copy queue length: 0  
Content index state: Healthy

[Resume | Update | Remove](#)

[View details](#)

Or resume the database copy using PowerShell.

```
[PS] C:\>Resume-MailboxDatabaseCopy DB01\SYDEX2
```

### *Reseeding a Failed Database Copy*

From time to time a database copy may fail. This could be due to database corruption, faulty storage, or a whole faulty server.

After the underlying cause has been resolved, such as replacing failed hardware, the database copy must be reseeded to that server. This is a similar process to adding a new database copy, in that the database and transaction log files are copied from another DAG member.

First the failed database copy is suspended.

```
[PS] C:\>Suspend-MailboxDatabaseCopy DB01\SYDEX2
```

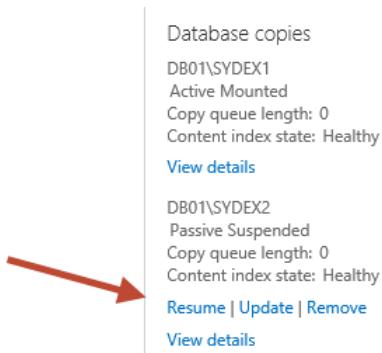
Next, reseed the database copy. If no additional switches are used then the reseed will use the current active database copy as the source. However you can optionally specify a source server, and in addition to that you can use a switch to make Exchange overwrite any existing database files that may still exist in the path where the database is located.

```
[PS] C:\>Update-MailboxDatabaseCopy DB01\SYDEX2 -SourceServer SYDEX1 -SafeDeleteExistingFiles
```

### *Removing a Database Copy*

The most common scenario in which a database copy will need to be removed from a DAG member is if the server itself is being uninstalled. Exchange Server 2013 cannot be uninstalled from a server while it is still a member of a DAG, and it cannot be removed from a DAG while it still hosts database copies.

A database copy can be removed using the Exchange Admin Center. After selecting a database click on **Remove** in the right-pane of the webpage.



Or remove the database copy using PowerShell.

```
[PS] C:\>Remove-MailboxDatabaseCopy DB01\SYDEX2
```

**Note:** This process cannot be used to remove the last copy of a mailbox database from a DAG. If the database is down to just one copy then it is removed using Remove-MailboxDatabase instead, which removes it entirely from the organization, just the same as when removing a database from a standalone mailbox server.

## Managing Database Switchovers

A switchover is an administrator-initiated move of the active database copy from one DAG member to another.

At a very basic level this process involves the active database copy dismounting on the current Mailbox server that is hosting it, and then mounting on the server that the active copy is being moved to.

Remember that each DAG member is hosting a copy of the database that is updated by continuous replication. So the switchover process can be quite fast, because one copy is dismounting and the other is mounting. The full database file doesn't need to be moved or copied between the DAG members at the time of the switchover.

Switchovers can be either targetless or targeted. In a targetless switchover the administrator issues the Move-ActiveMailboxDatabase cmdlet but does not specify which DAG member it should switchover to. Instead, best copy and server selection (BCSS) is allowed to choose the database copy that should become the active copy.

```
[PS] C:\> Move-ActiveMailboxDatabase DB01
```

Identity	ActiveServerAtS	ActiveServerAtE	Status	NumberOfLogsLost
DB01	sydex2	sydex1	Succeeded	0

A single command can also be used to move all active database copies off a server.

```
[PS] C:\> Move-ActiveMailboxDatabase -Server SYDEX1
```

Identity	ActiveServerAtS	ActiveServerAtE	Status	NumberOfLogsLost
DB01	sydex1	sydex2	Succeeded	0
DB03	sydex1	sydex2	Succeeded	0

In a targeted switchover the administrator specifies the server where the database copy should become active.

```
[PS] C:\> Move-ActiveMailboxDatabase DB01 -ActivateOnServer SYDEX2
```

Identity	ActiveServerAtS	ActiveServerAtE	Status	NumberOfLogsLost
DB01	sydex1	sydex2	Succeeded	0

**Real World:** Some administrators are nervous about performing database switchovers during business hours, in case something goes wrong and there is a service interruption.

This is somewhat misguided, because if you don't trust your DAG to switchover correctly then how can you trust it to failover correctly when a fault occurs? If your switchovers are causing problems then it would be worth revisiting your DAG design.

## Database Failovers

Failovers are unplanned events that result in the active database copy moving to another DAG member. Situations such as a server having a storage failure or crashing completely will cause database failovers to occur.

There are also a variety of server health conditions that may result in a database failover being initiated by Managed Availability as it attempts to avoid an incident or restore service.. You can read more about Managed Availability in the Managing and Monitoring chapter of this guide.

When a failover occurs there are log entries written to the event logs on all of the DAG members, which aids in troubleshooting problems when a server goes completely offline because log data is available to examine on other servers, rather than all of the log data going offline along with the server that failed.

In this example, database DB01 was active on server SYDEX1 when the server crashed due to a hardware failure. The event log on SYDEX2 shows the detection of the failure and the recovery actions taken.

Level	Date and Time	Source	Event ID	Task Category
Information	14/07/2014 2:22:22 PM	HighAvailability	322	Database Action
Information	14/07/2014 2:22:22 PM	HighAvailability	315	Database Action
Information	14/07/2014 2:22:22 PM	HighAvailability	314	Database Action
Warning	14/07/2014 2:22:22 PM	HighAvailability	107	Database Action
Information	14/07/2014 2:22:22 PM	HighAvailability	336	Database Action
Error	14/07/2014 2:22:22 PM	HighAvailability	163	Database Action
Warning	14/07/2014 2:22:22 PM	HighAvailability	331	Database Action
Warning	14/07/2014 2:22:21 PM	HighAvailability	324	Database Action
Error	14/07/2014 2:22:21 PM	HighAvailability	711	Dumpster Redelivery
Information	14/07/2014 2:22:15 PM	HighAvailability	218	Database Action
Information	14/07/2014 2:22:15 PM	HighAvailability	726	Database Action
Error	14/07/2014 2:22:15 PM	HighAvailability	312	Database Action
Error	14/07/2014 2:22:15 PM	HighAvailability	320	Database Action

We'll look at this in more detail in the section of this chapter on best copy and server selection in action.

## Configuring DAC Mode

The current setting for Datacenter Activation Coordination Mode can be viewed using `Get-DatabaseAvailabilityGroup`.

```
[PS] C:\>Get-DatabaseAvailabilityGroup | Select Name,*mode*
Name DatacenterActivationMode
-----
DAG1 Off
```

Use `Set-DatabaseAvailabilityGroup` to enable DAC Mode.

```
[PS] C:\>Set-DatabaseAvailabilityGroup DAG1 -DatacenterActivationMode DagOnly
```

## Extending DAGs to Multiple Sites

The members of a database availability group can exist in multiple sites (or datacenters) within the organization's network.

Extending a DAG to multiple sites in this way can maintain service availability even when an entire datacenter goes offline. Of course, this also depends on other elements of the network such as Active Directory, DNS, internet connectivity, WAN connectivity between clients and servers, and so on, also being available when one datacenter has gone offline.

## One DAG or Two for Site Resilience?

When a DAG is extended to multiple sites an organization usually has a preference about how the DAG runs, either:

- Active-Passive, where all active database copies are in one datacenter (the “primary” datacenter), and the other only hosts passive database copies unless the primary datacenter fails.
- Active-Active, where active database copies can be hosted in either datacenter location.

In some cases active-passive is preferred because the organization wants end users to connect to one particular datacenter most of the time, but with multiple client locations there may be some that are better connected to one datacenter or another. In this situation an organization may choose to implement two active-passive DAGs, with each having a different primary datacenter that is closest to those mailbox users that are hosted on its databases.

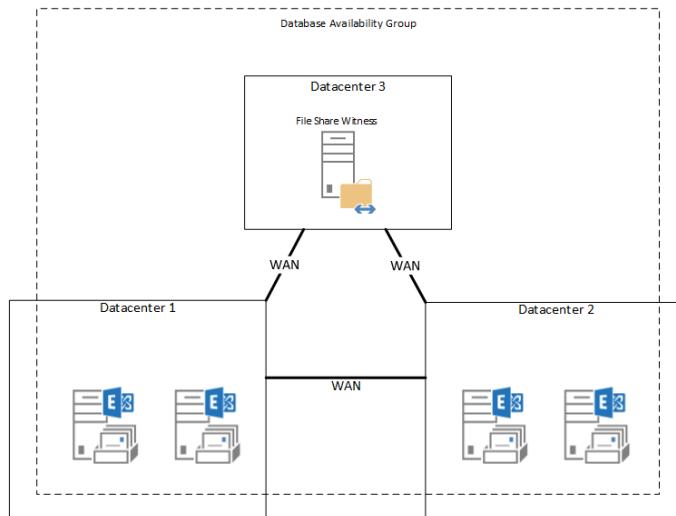
## Placing a File Share Witness in a Third Site

Earlier in this chapter we looked at the concept of quorum, and also looked at how DAC Mode helps prevent split brain scenarios particularly for multi-site DAGs.

In that example the file share witness for the DAG was hosted in the primary datacenter, and became unavailable when the datacenter went offline. To restore service in that situation manual action is required by the administrator.

When the file share witness is placed in a separate, third datacenter location that is independently connected to the other datacenters hosting DAG members, manual service recovery steps may not be required at all.

This is because the file share witness is still available to form quorum with the remaining DAG members in the datacenter that is still online.

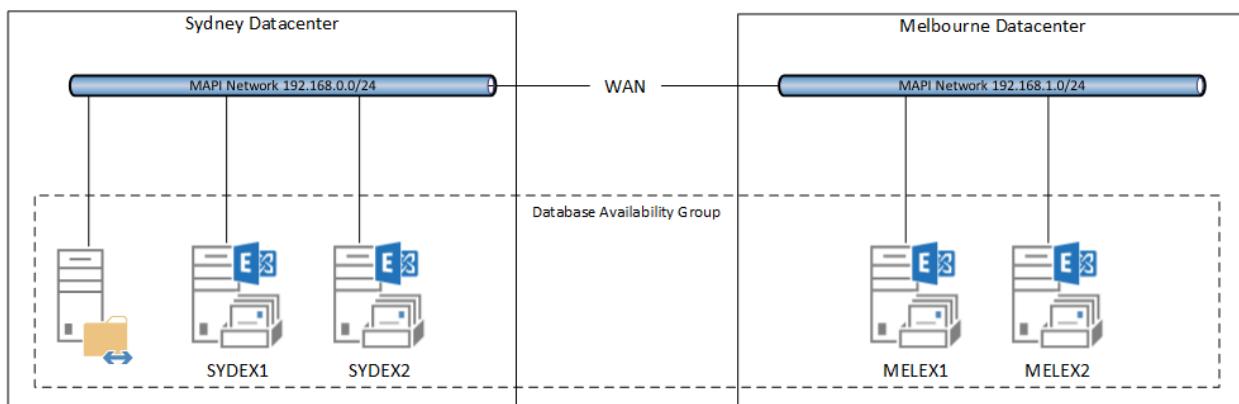


The network connectivity to the third datacenter is key to this solution working and being supported. If connectivity to the file share witness was lost when one of the datacenters went offline then quorum could not be maintained. A full “mesh” network with independent connectivity between each datacenter is a requirement when placing the FSW in a third datacenter.

**Note:** A natural assumption would be to use Microsoft Azure as the third location to host the file share witness. However, this is not currently supported due to Azure not being able to establish a VPN connection to more than one destination. But with high demand for this capability from customers this scenario is likely to be supported in the near future.

## Demonstration: Site Failover for Disaster Recovery

Let's take a look at a site failover scenario. In this example the DAG is running across two datacenters in Sydney and Melbourne, with two DAG members in each datacenter and the file share witness in the Sydney datacenter as the “primary” location.



All databases are currently mounted and operating normally, evenly distributed between the four DAG members.

Test-MAPICConnectivity indicates that all databases are online and able to service client connections.

[PS] C:\>Get-MailboxDatabase   Test-MAPICConnectivity			
MailboxServer	Database	Result	Error
-----	-----	-----	-----
SYDEX2	DB01	Success	
SYDEX2	DB02	Success	
SYDEX2	DB03	Success	
SYDEX2	DB04	Success	

And Get-MailboxDatabaseCopyStatus tells us that the active database copies are distributed evenly across the four available DAG members.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * -Active | select name,status,mailboxserver  
Name      Status MailboxServer  
----  
DB01\SYDEX1 Mounted SYDEX1  
DB02\SYDEX2 Mounted SYDEX2  
DB03\MELEX1 Mounted MELEX1  
DB04\MELEX2 Mounted MELEX2
```

The Sydney datacenter suffers a failure and both Sydney DAG members along with the Sydney file share witness go offline. The cluster loses quorum, and the DAG goes offline.

```
[PS] C:\>Get-MailboxDatabase | Test-MAPICConnectivity  
MailboxServer    Database    Result    Error  
-----  
SYDEX1          DB01        *FAILURE* The transaction didn't complete in the allotted time  
                           (60 seconds).  
SYDEX2          DB02        *FAILURE* The transaction didn't complete in the allotted time  
                           (60 seconds).  
MELEX1          DB03        *FAILURE* Database is dismounted.  
MELEX2          DB04        *FAILURE* Database is dismounted.
```

The aim of the recovery process is to bring the DAG online again with only the two Melbourne DAG members available, and an alternate file share witness server. DAC mode was already enabled for this DAG because, as discussed earlier in this chapter, it is recommended to enable DAC mode for all DAGs with two or more members, especially multi-site DAGs.

First, run Stop-DatabaseAvailabilityGroup cmdlet to inform the surviving DAG members in the Melbourne site that the Sydney site servers are unavailable.

```
[PS] C:\>Stop-DatabaseAvailabilityGroup DAG1 -ActiveDirectorySite Sydney -ConfigurationOnly  
Confirm  
Are you sure you want to perform this action?  
Stopping Mailbox servers for Active Directory site "Sydney" in database availability group "DAG1".  
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
```

Next, stop the Cluster service on each Melbourne DAG member.

```
[PS] C:\>Invoke-Command -ComputerName MELEX1,MELEX2 {Stop-Service ClusSvc}
```

Now we can run `Restore-DatabaseAvailabilityGroup` to identify the DAG members in Melbourne that are being used to restore service, and configure the alternate witness server.

```
[PS] C:\>Restore-DatabaseAvailabilityGroup DAG1 -ActiveDirectorySite Melbourne -AlternateWitnessServer MELDC1

Confirm
Are you sure you want to perform this action?
Restoring Mailbox servers for Active Directory site "Melbourne" in database availability group "DAG1".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
```

At this stage the mailbox databases will be mounted automatically, unless there are activation blocks on the DAG members in the Melbourne site, or activation blocks on the database copies hosted on the Melbourne DAG members.

We can verify the status of the databases with `Get-MailboxDatabaseCopyStatus`.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * -Active | select name,status,mailboxserver

Name      Status MailboxServer
----      -----
DB01\MELEX1 Mounted MELEX1
DB02\MELEX1 Mounted MELEX1
DB03\MELEX1 Mounted MELEX1
DB04\MELEX2 Mounted MELEX2
```

Another test worth running is the `Test-MapiConnectivity` cmdlet.

```
[PS] C:\>Get-MailboxDatabase | Test-MAPICConnectivity

MailboxServer    Database    Result    Error
-----          -----      -----      -----
MELEX1          DB01        Success
MELEX1          DB02        Success
MELEX1          DB03        Success
MELEX2          DB04        Success
```

If any such activation blocks do exist you can either:

1. Manually activate the database copy using `Move-ActiveMailboxDatabase`. Activation blocks only prevent automatic activation, not manual activation like this.
2. Remove the activation block for the mailbox server.
3. Remove the activation block for the database copy.

For example, to remove the activation block from a mailbox server use `Set-MailboxServer`.

```
[PS] C:\>Set-MailboxServer MELEX2 -DatabaseCopyAutoActivationPolicy Unrestricted
```

Or to remove the activation block from a database copy use `Resume-MailboxDatabaseCopy`.

```
[PS] C:\>Resume-MailboxDatabaseCopy DB04\MELEX2
```

Of course, datacenter failover scenarios like this don't just involve the Mailbox servers. There are also Client Access and Transport services to take into consideration.

For example, if you are using DNS round robin you may need to remove the DNS records for the Client Access namespaces that point to Sydney servers or load balancers so that clients no longer attempt to connect to those failed servers.

You may also need to adjust or configure your Send Connectors and inbound SMTP routing so that internet email flow continues to work.

For now let's focus primarily on the Mailbox servers. When the Sydney datacenter is back online there are further steps required to restore the DAG to its original condition.

The Sydney DAG members are still considered to be "stopped" or inactive.

```
[PS] C:\>Get-DatabaseAvailabilityGroup DAG1 -Status | select *servers*
```

Servers	:	{MELEX1, MELEX2, SYDEX2, SYDEX1}
StoppedMailboxServers	:	{SYDEX1.exchange2013demo.com, SYDEX2.exchange2013demo.com}
StartedMailboxServers	:	{MELEX1.exchange2013demo.com, MELEX2.exchange2013demo.com}
OperationalServers	:	{MELEX2, MELEX1}

Running `Start-DatabaseAvailabilityGroup` for the Sydney AD site will make those DAG members operational again.

```
[PS] C:\>Start-DatabaseAvailabilityGroup DAG1 -ActiveDirectorySite Sydney
```

Now the Sydney DAG members are considered operational.

```
[PS] C:\>Get-DatabaseAvailabilityGroup DAG1 -Status | select *servers*
```

Servers	:	{MELEX1, MELEX2, SYDEX2, SYDEX1}
StoppedMailboxServers	:	{}
StartedMailboxServers	:	{SYDEX1.exchange2013demo.com, SYDEX2.exchange2013demo.com, MELEX1.exchange2013demo.com, MELEX2.exchange2013demo.com}
OperationalServers	:	{SYDEX2, SYDEX1, MELEX2, MELEX1}

There may be some time required to allow the database copies in Sydney to resynchronize. It is also possible that some database copies in Sydney would need to be reseeded. Wait until all database replication is healthy again before proceeding further.

Name	Status	CopyQueue Length	ReplayQueue Length
DB01\SYDEX1	Healthy	0	0
DB03\SYDEX1	Healthy	0	0
DB04\SYDEX1	Healthy	0	0
DB02\SYDEX1	Healthy	0	267
DB02\SYDEX2	Healthy	0	0
DB01\SYDEX2	Healthy	0	0
DB03\SYDEX2	Healthy	0	8
DB04\SYDEX2	Healthy	0	0
DB01\MELEX1	Mounted	0	0
DB02\MELEX1	Mounted	0	0
DB03\MELEX1	Mounted	0	0
DB04\MELEX1	Healthy	0	511
DB01\MELEX2	Healthy	0	125
DB02\MELEX2	Healthy	0	0
DB03\MELEX2	Healthy	0	0
DB04\MELEX2	Mounted	0	0

The DAG is currently using the alternate file share witness that was configured during the recovery process in Melbourne. To revert to the primary file share witness run Set-DatabaseAvailabilityGroup with no additional parameters.

```
[PS] C:\>Set-DatabaseAvailabilityGroup DAG1
```

Finally, when you are satisfied that everything is healthy and functioning correctly, you can switchover databases to the Sydney datacenter again, or rebalance the DAG using RedistributeActiveDatabases.ps1.

```
[PS] C:\>cd $exscripts  
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>.\RedistributeActiveDatabases.ps1 -DagName DAG1 -BalanceDbsByActivationPreference -Confirm:$false
```

# Lagged Database Copies

A lagged database copy is a passive database copy where the transaction log data is not immediately committed to that database copy.

The lag interval is configurable by the administrator and specifies the amount of time between when a transaction log file is generated and when it is replayed into the passive database copy. The default lag interval is 0 and the maximum lag interval is 14 days.

The purpose of a lagged copy is to provide the ability to recover the database from an earlier point in time if some kind of database fault occurs, such as logical corruption.

Lagged copies were first introduced in Exchange Server 2010, and have some improvements in Exchange Server 2013. For example, the lagged copy can perform automatic log play down for a variety of situations, such as when a database page fault has been detected and needs patching, when a low free disk space condition is detected, or when the lagged copy is the only available copy of a database for a period of time.

## Configuring Lagged Database Copies

The replay lag interval for a lagged database copy is configured using `Set-MailboxDatabaseCopy`, and the value is in the format “days.hours:minutes:seconds”. For example, a value of “7.0:0:0” means 7 days.

It is common to choose the least preferred database copy to be the lagged copy, so in our demo environment with four database copies the copy with activation preference of 4 would be set as the lagged copy.

To select all database copies that have an activation preference of 4 we can use `Get-MailboxDatabaseCopyStatus`.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | where {$_.ActivationPreference -eq "4"}  
Name      Status CopyQueueLength ReplayQueueLength LastInspectedLogTime ContentIndexState  
----      ----  -----  -----  -----  
DB02\SYDEX1 Healthy 0          0          23/09/2014 10:53:37 PM Healthy  
DB03\SYDEX2 Healthy 0          0          23/09/2014 10:45:59 PM Healthy  
DB04\MELEX1 Healthy 0          0          23/09/2014 10:47:08 PM Healthy  
DB01\MELEX2 Healthy 0          0          23/09/2014 10:45:02 PM Healthy
```

Piping that output to `Set-MailboxDatabaseCopy` will configure the replay lag interval.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | where {$_.ActivationPreference -eq "4"} | Set-MailboxDatabaseCopy -ReplayLagTime 7.0:0:0
```

**Real World:** When you set the replay lag time you will see a warning about the SafetyNetHoldTime as well. It is always recommended to set Safety Net hold time to the same value or greater value than the replay lag time.

In addition to setting the replay lag time you can also prevent a lagged database copy from being automatically activated during a database failover scenario. To block automatic activation for a database copy use the Suspend-MailboxDatabaseCopy cmdlet.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | where {$_.ActivationPreference -eq "4"} | Suspend-MailboxDatabaseCopy -ActivationOnly
```

When automatic activation has been blocked in this manner the database copy can only become active after manual intervention by an administrator.

**Real World:** Blocking automatic activation for a database copy reduces the number of database copies that are available for Active Manager to attempt to mount during the Best Copy and Server Selection process.

It is a decision for your organization whether the preservation of the lagged database copy by blocking automatic activation is more important than having the maximum number of available database copies available to mount in a failover scenario.

## Using Lagged Database Copies in Recovery Scenarios

Lagged copies can be used for recovery in a variety of ways.

- Activate a lagged database copy by replaying all uncommitted log files. In this scenario a decision is made to replay all of the log files in the replay queue into the database and bring it online.
- Activate a lagged database copy to a specific point in time. In this scenario some of the log files are removed from the server to prevent them from being replayed into the database copy as it is brought online.
- Activate a lagged database copy and use Safety Net for recovering lost data. In this scenario the database is mounted without replaying transaction log from the replay queue, and messages are resubmitted to the database from Safety Net.

In each of these scenarios there is an optional (but recommended) step to make a copy of the lagged database files before activating it, so that you are still able to use the lagged copy at a later stage if you have a different recovery scenario arise.

### *Activating a Lagged Database Copy by Replaying All Log Files*

A lagged database copy can be activated by replaying all of the uncommitted transaction log files (those that are in the replay queue) into the database and then mounting it. This effectively “catches up” the database copy to the others.

The amount of time it takes to commit all of the outstanding transaction log files to the database will depend on the length of the lag interval and the amount of transaction log data that has been generated in that space of time. For a busy Exchange environment with a lag interval of 7 days this could take several hours.

First, identify the lagged copy of the database. In this example of database DB01 the lagged copy is DB01\MELEX2.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus DB01 | Where {$_.ReplayLagStatus.Enabled -eq $true}

Name      Status  CopyQueueLength ReplayQueueLength LastInspectedLogTime  ContentIndexState
----      -----  -----          -----          -----          -----
DB01\MELEX2 Healthy 0           361           26/09/2014 11:46:31 PM Healthy
```

Suspend the lagged database copy using Suspend-MailboxDatabaseCopy.

```
[PS] C:\>Suspend-MailboxDatabaseCopy DB01\MELEX2
```

Make a backup of the database and log files for this database copy. You can use a backup application or simply copy them to another location.

A suspended database copy can't be made active, so we need to resume the lagged database copy again. It may take a few minutes afterwards for the content index to return to a healthy state as well.

```
[PS] C:\>Resume-MailboxDatabaseCopy DB01\MELEX2
```

Finally, activate the lagged database copy. Note that this uses the same Move-ActiveMailboxDatabase cmdlet as a normal database switchover, with the additional –SkipLagChecks switch.

```
[PS] C:\>Move-ActiveMailboxDatabase DB01 -ActivateOnServer MELEX2 -SkipLagChecks
```

When all of the log files have been replayed the database copy will mount and become the active database copy.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus DB01\MELEX2
Name      Status CopyQueueLength ReplayQueueLength LastInspectedLogTime ContentIndexState
----      ----- ----- ----- ----- -----
DB01\MELEX2 Mounted 0          0          Healthy
```

The database copy still has the same replay lag interval configured, it is just not in effect as long as the database copy is active. If the active database copy is moved to another DAG member then the replay lag interval will come into effect again and the replay queue length will begin increasing for the lagged database copy.

#### *Activating a Lagged Database Copy to a Point in Time*

Activating a lagged database copy to a specific point in time follows a different process, and is often used as a way to make a copy of the database to mount as a recovery database, rather than activate the lagged copy itself.

First, identify the lagged copy of the database. In this example of database DB02 the lagged copy is DB02\SYDEX1, so this procedure should be performed on server SYDEX1.

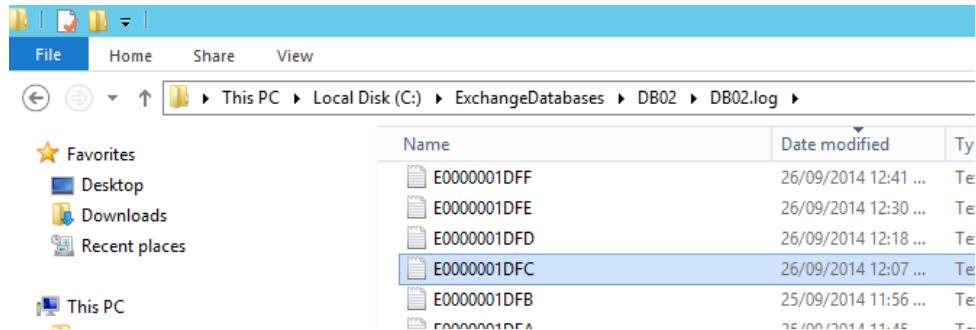
```
[PS] C:\>Get-MailboxDatabaseCopyStatus DB02 | Where {$_.ReplayLagStatus.Enabled -eq $true} | ft -auto
Name      Status CopyQueueLength ReplayQueueLength LastInspectedLogTime ContentIndexState
----      ----- ----- ----- ----- -----
DB02\SYDEX1 Healthy 0          391          27/09/2014 12:16:08 AM Healthy
```

Suspend the lagged database copy using Suspend-MailboxDatabaseCopy.

```
[PS] C:\>Suspend-MailboxDatabaseCopy DB02\SYDEX1
```

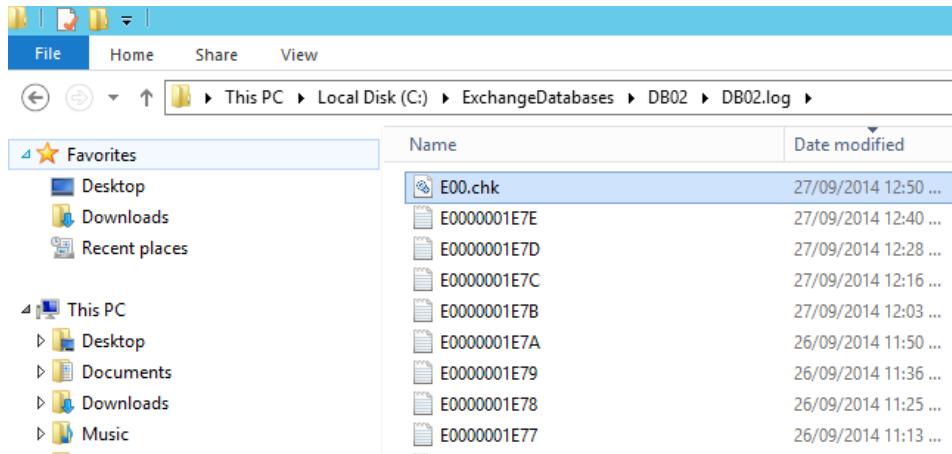
Make a backup of the database and log files for this database copy. You can use a backup application or simply copy them to another location.

Look at the log file timestamps to determine which log files were created after the point in time you want to recover to.



Move the log files after the time that you're recovering to into another folder. Don't delete them, just in case you need them, but they do need to be moved to another folder.

Delete the .chk file from the transaction log folder.



Using ESEUtil in a CMD prompt we can see that the database file is in a dirty shutdown state.

```
C:\>eseutil /mh C:\ExchangeDatabases\DB02\DB02.db\DB02.edb

Extensible Storage Engine Utilities for Microsoft(R) Exchange Server
Version 15.00
Copyright (C) Microsoft Corporation. All Rights Reserved.

Initiating FILE DUMP mode...
Database: C:\ExchangeDatabases\DB02\DB02.db\DB02.edb

DATABASE HEADER:
Checksum Information:
Expected Checksum: 0xe4311492
Actual Checksum: 0xe4311492

Fields:
State: Dirty Shutdown
```

So the next step is to perform a soft recovery using the log files that we've left in the log folder.

Navigate to the log folder for the database and run ESEUtil with the /r switch and the log prefix (E00 in this example).

```
C:>>cd ExchangeDatabases\DB02\DB02.log
C:\ExchangeDatabases\DB02\DB02.log>eseutil /r e00 /a

Extensible Storage Engine Utilities for Microsoft(R) Exchange Server
Version 15.00
Copyright (C) Microsoft Corporation. All Rights Reserved.

Initiating RECOVERY mode...
    Logfile base name: e00
        Log files: <current directory>
        System files: <current directory>

Performing soft recovery...
                    Restore Status (% complete)

    0   10   20   30   40   50   60   70   80   90   100
    |-----|-----|-----|-----|-----|-----|-----|-----|
    ..... .

Operation completed successfully in 31.859 seconds.
```

Check the database file again to verify it is in a clean shutdown state.

```
C:>eseutil /mh C:\ExchangeDatabases\DB02\DB02.db\DB02.edb

Extensible Storage Engine Utilities for Microsoft(R) Exchange Server
Version 15.00
Copyright (C) Microsoft Corporation. All Rights Reserved.

Initiating FILE DUMP mode...
    Database: C:\ExchangeDatabases\DB02\DB02.db\DB02.edb

DATABASE HEADER:
Checksum Information:
Expected Checksum: 0xf3f5ae3e
Actual Checksum: 0xf3f5ae3e

Fields:
    State: Clean Shutdown
```

The database file can now be copied to another location and used as a recovery database for a restore operation.

The lagged database copy itself can be resumed as well. If you wanted to return it to the lagged state before the recovery process above was run you would need to restore the files you backed up at the start of this process before resuming the database copy.

```
[PS] C:\>Resume-MailboxDatabaseCopy DB02\SYDEX1
```

### *Activating a Lagged Database Copy Using Safety Net Recovery*

Activating a lagged database copy using Safety Net recovery involves removing all unnecessary transaction log files from the log folder, so that only the bare minimum required for mounting the database still exist.

First, identify the lagged copy of the database. In this example of database DB03 the lagged copy is DB02\SYDEX2, so this procedure should be performed on the server SYDEX2.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus DB03 | Where {$_.ReplayLagStatus.Enabled -eq $true} |  
Name      Status  CopyQueueLength ReplayQueueLength LastInspectedLogTime ContentIndexState  
----      -----  -----          -----          -----          -----  
DB03\SYDEX2 Healthy 0           379           27/09/2014 1:11:18 AM Healthy
```

Suspend the lagged database copy using Suspend-MailboxDatabaseCopy.

```
[PS] C:\>Suspend-MailboxDatabaseCopy DB03\SYDEX2
```

Make a backup of the database and log files for this database copy. You can use a backup application or simply copy them to another location.

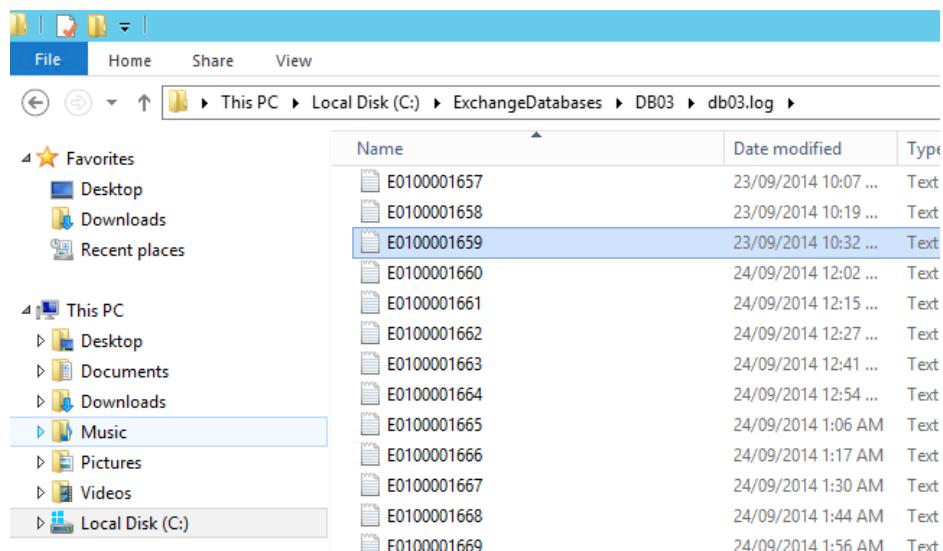
Use ESEUtil to determine which log files are required by the database.

```
C:\>eseutil /mh C:\ExchangeDatabases\DB03\DB03.db\DB03.edb  
Extensible Storage Engine Utilities for Microsoft(R) Exchange Server  
Version 15.00  
Copyright (C) Microsoft Corporation. All Rights Reserved.  
  
Initiating FILE DUMP mode...  
Database: C:\ExchangeDatabases\DB03\DB03.db\DB03.edb  
  
DATABASE HEADER:  
Checksum Information:  
Expected Checksum: 0x62a178a2  
Actual Checksum: 0x62a178a2  
  
Fields:  
Log Required: 5722-5728 (0x165a-0x1660)
```

There are two hexadecimal values displayed in the ESEUtil output. In this example they are “0x165a” and “0x1660”. It is the second value that is of interest, because this is the “High Generation” number.

Move all log files that have a sequence number that is higher than the “High Generation” number. For example, 0x1660 means log files E0100001661 and above should be removed from the server hosting the lagged copy, which is SYDEX2.

Again don’t delete the files, just move them to a temporary folder in case you need them again.



**Note:** Transaction log file sequence numbers use a hexadecimal numbering sequence. This means that, for example, log file E0100001699 is not followed by E0100001700, but rather E010000169A instead. The last log file in the E01000016xx range is E01000016FF, and then comes E0100001700.

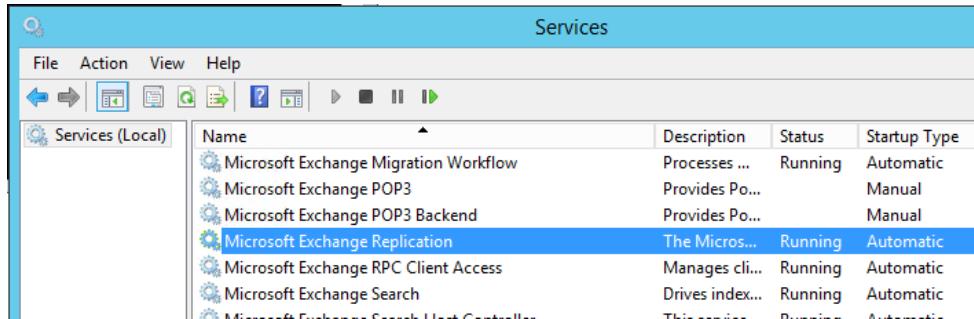
This is important to be aware of because sorting log files by name in Windows Explorer does not correctly sort them by log file sequence number. If you are not aware of this, and do not remove all of the correct log files, then the switchover to the lagged database copy will fail.

Next, identify the server hosting the active copy of the database. In this example it is the server SYDEX1.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus DB03 -Active | ft -auto
```

Name	Status	CopyQueueLength	ReplayQueueLength	LastInspectedLogTime	ContentIndexState
DB03\SYDEX1	Mounted	0			Healthy

On the server hosting the active copy of the database stop the Microsoft Exchange Replication service



Name	Description	Status	Startup Type
Microsoft Exchange Migration Workflow	Processes ...	Running	Automatic
Microsoft Exchange POP3	Provides Po...	Manual	Manual
Microsoft Exchange POP3 Backend	Provides Po...	Manual	Manual
Microsoft Exchange Replication	The Micros...	Running	Automatic
Microsoft Exchange RPC Client Access	Manages cli...	Running	Automatic
Microsoft Exchange Search	Drives index...	Running	Automatic

Perform a database switchover to activate the lagged database copy, in this example the server is SYDEX2.

```
[PS] C:\>Move-ActiveMailboxDatabase DB03 -ActivateOnServer SYDEX2 -MountDialOverride BestEffort -SkipActiveCopyChecks -SkipClientExperienceChecks -SkipHealthChecks -SkipLagChecks
```

The database will mount and make a request to Safety Net for resubmission of the mail that is missing from the database.

## After Lagged Copy Activation

Activating a lagged database copy is not the end of the exercise. There are still several things that you should do to ensure that your environment is back to a healthy state. After all, some event occurred to cause you to activate a lagged copy, so you most likely still have work ahead of you to restore your environment to its original healthy condition.

Here are some steps that you should consider:

- Use Get-MailboxDatabaseCopyStatus to review the health of your database copies
- Re-seed any database copies that are failed or otherwise considered to be unusable (eg, if the reason you activated a lagged copy was due to corruption in your database)
- Move the active database copy back to a more preferred server
- Re-apply or verify that your lagged copy configuration is in effect

## A Deeper Look at DAG Features

We've discussed a lot of concepts and features of Exchange Server 2013 Mailbox server high availability so far. In this section let's take a deeper look at a few of those concepts and features in action.

In particular these are concepts and features that you are likely to encounter in day to day operations for Exchange Server 2013 if a failure occurs.

## Best Copy and Server Selection in Action

Database copies that are currently unreachable or have been configured by the administrator to be blocked from activation are not included as failover candidates in the BCSS process.

Database copies that are considered failover candidates are sorted depending on a number of variables.

The first is the AutoDatabaseMountDial setting, which is configured on each Mailbox server in the DAG. This setting configures the threshold for the number of missing log files (or the copy queue length) for the database copy for it to be considered unmountable. There are three possible settings:

- GoodAvailability – this is the default setting and allows a database copy to be automatically mounted if it has a copy queue length of six or less.
- BestAvailability – allows a database copy to be automatically mounted if it has a copy queue length of 12 or less.
- Lossless – allows a database copy to be automatically mounted only if there are no missing log files.

**Note:** It may seem like “Lossless” is the only logical choice here, because who would willingly choose an option that may result in data loss? However it is often better to allow service to be restored by mounting a database copy that has some missing log files, and then data loss can be mitigated by other means such as by requesting email messages to be resubmitted from Safety Net.

When AutoDatabaseMountDial is set to Lossless on any DAG member, the failover candidates are sorted in ascending order of activation preference.

When AutoDatabaseMountDial is set to GoodAvailability or BestAvailability on all DAG members, the failover candidates are sorted by their copy queue length, in order of shortest to longest. If two database copies have the same copy queue length then the activation preference is used as the tie breaker.

**Real World:** The recommended practice is to set the AutoDatabaseMountDial setting to be the same for all members of a DAG.

At this stage in the process even database copies that have missing log files in excess of the AutoDatabaseMountDial threshold are still in consideration as a potential activation candidate, because a process called “attempt copy last logs” (ACLL) will run to try and copy the missing log files from the server that last hosted the active database copy before the failure occurred.

ACLL is useful in situations where the failure is not a total server failure, and the log files are still accessible, which allows a passive database copy that may be excluded from consideration as a failover candidate due to missing log files to copy the missing files and therefore get itself back into consideration.

However, before ACLL runs the best copy selection process needs to choose a database copy to be the candidate to become the active database copy. For that to occur a series of checks is performed on the activation candidates, that includes the following characteristics:

- Copy queue length – the number of transaction log files yet to be shipped from the server hosting the active database copy
- Replay queue length – the number of transaction log files that have been shipped to the passive copy's server but have not yet been replayed into the passive database copy
- Database status – such as “Healthy” or “Failed”, among other possible values
- Content index status – such as “Healthy” or “Failed”, among other possible values

Exchange Server 2013 also leverages Managed Availability so that server health is also taken into consideration for best copy selection (which is why in Exchange Server 2013 it is also called “best copy and server selection”).

The server health checks performed are (in order):

1. All Healthy – Active Manager looks for a DAG member hosting a copy of the database that has all server components in a state of Healthy.
2. Up to Normal Healthy – Active Manager looks for a DAG member hosting a copy of the database that has all server components with Normal priority in a healthy state.
3. All Better than Source – Active Manager looks for a DAG member hosting a copy of the database that has server components in a better state than the current active server.
4. Same as Source – Active Manager looks for a DAG member hosting a copy of the database that has server components in the same state as the current active server.

When BCSS is run as a result of a Managed Availability triggered failover, the health of the target server's component must be better than the health of the component on the active server. For example, if the active server has a failure of ActiveSync, then BCSS will not select another DAG member that also has an unhealthy ActiveSync component.

You can read more about server health states in the Managing and Monitoring chapter of this guide.

Beginning with the list of activation candidates sorted according to the AutoDatabaseMountDial setting, Active Manager will look for a database copy to mount.

The criteria sets will be used by Active Manager to assess the list of activation candidates in order. For example, each activation candidate is assessed against criteria set #1.

If after looking at each activation candidate in the list Active Manager does not finding a suitable database copy to mount, it will move on to criteria set #2 and assess each activation candidate in the list again, and so on until all criteria sets have been used.

Criteria Set	Database Copy Status	Other Status
<b>1</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Healthy</li> <li>Copy queue length less than 10</li> <li>Replay queue length less than 50</li> </ul>
<b>2</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Crawling</li> <li>Copy queue length less than 10</li> <li>Replay queue length less than 50</li> </ul>
<b>3</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Healthy</li> <li>Replay queue length less than 50</li> </ul>
<b>4</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Crawling</li> <li>Replay queue length less than 50</li> </ul>
<b>5</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Replay queue length less than 50</li> </ul>
<b>6</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Healthy</li> <li>Copy queue length less than 10</li> </ul>
<b>7</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Crawling</li> <li>Copy queue length less than 10</li> </ul>
<b>8</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Healthy</li> </ul>
<b>9</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	<ul style="list-style-type: none"> <li>Content Index status of Crawling</li> </ul>
<b>10</b>	Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource	n/a

As you can see if Active Manager gets all the way to criteria set #10 without finding a suitable database copy to mount, then it will attempt to mount any database copy that is Healthy, DisconnectedAndHealthy, DisconnectedAndResynchronizing, or SeedingSource.

If no database copies meet that final criteria set then Active Manager will not automatically activate any of the database copies, and manual action by an administrator will be required to restore service.

**Real World:** Best copy selection runs every time a “targetless” database switchover or failover (one where an administrator hasn’t selected a database copy to become active) occurs within the database availability group.

However for a lot of environments the DAG is healthy enough that BCS will not get past criteria set #1 before finding a suitable database copy to mount. In fact, when the default AutoDatabaseMountDial setting of GoodAvailability is used, BCS will often choose the database copy that is next in order of activation preference, because the activation preference value was used as a tie breaker between multiple database copies with equal copy queue lengths (often 1 or 0 in a healthy environment).

This is why many administrators simply assume that activation preference is the sole determining factor for which database copy becomes active in a failover scenario.

## Dynamic Quorum in Action

Let’s take a closer look at how Dynamic Quorum helps to keep an Exchange Server 2013 DAG online during multiple server failures.

To begin with here are the DAG members, all online, and their respective values for NodeWeight and DynamicWeight.

```
PS C:\> Get-ClusterNode | Select name,state,nodeweight,dynamicweight
Name   State NodeWeight DynamicWeight
----  -----
MELEX1 Up      1          1
MELEX2 Up      1          1
SYDEX1 Up      1          1
SYDEX2 Up      1          1
```

Let’s assume that the Melbourne datacenter has experienced a power loss and both Melbourne DAG members have gone offline as a result.

The DAG itself remains online only because the Sydney nodes plus the file share witness in Sydney are still able to form a majority (3 out of 5 votes) and achieve quorum.

But notice in the output below how the DynamicWeight has been adjusted to account for the two nodes that are offline.

The failed nodes have been removed from the voting process, reducing the number of voting members to 3 (2 DAG members and 1 file share witness).

```
PS C:\> Get-ClusterNode | Select name,state,nodeweight,dynamicweight

Name      State NodeWeight DynamicWeight
----      ---- -----
MELEX1    Down     1          0
MELEX2    Down     1          0
SYDEX1    Up       1          1
SYDEX2    Up       1          1
```

If we then lose one of the Sydney nodes to another failure, we've only lost 1 out of 3 required votes, and quorum is still maintained with the 2 remaining votes (1 DAG member and 1 file share witness).

```
PS C:\> Get-ClusterNode | Select name,state,nodeweight,dynamicweight

Name      State NodeWeight DynamicWeight
----      ---- -----
MELEX1    Down     1          0
MELEX2    Down     1          0
SYDEX1    Up       1          1
SYDEX2    Down     1          1
```

Without Dynamic Quorum if a four member DAG had three members offline it would not be able to achieve quorum with the remaining two out of five votes, and the DAG would go offline dismounting all databases and causing mailboxes to become unavailable for end users.

However, thanks to Dynamic Quorum this DAG has survived even down to the “last man standing”, and database copies are still active and mounted on the sole remaining DAG member.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | sort name | ft -auto

Name      Status      CopyQueueLength ReplayQueueLength LastInspectedLogTime ContentIndexState
----      ----
DB01\MELEX1 ServiceDown 0          0          Unknown
DB01\MELEX2 ServiceDown 0          0          Unknown
DB01\SYDEX1 Mounted     0          0          Healthy
DB01\SYDEX2 ServiceDown 0          0          Unknown
DB02\MELEX1 ServiceDown 0          0          Unknown
DB02\MELEX2 ServiceDown 0          0          Unknown
DB02\SYDEX1 Mounted     0          0          Healthy
DB02\SYDEX2 ServiceDown 0          0          Unknown
DB03\MELEX1 ServiceDown 0          0          Unknown
DB03\MELEX2 ServiceDown 0          0          Unknown
DB03\SYDEX1 Mounted     0          0          Healthy
DB03\SYDEX2 ServiceDown 0          0          Unknown
DB04\MELEX1 ServiceDown 0          0          Unknown
DB04\MELEX2 ServiceDown 0          0          Unknown
DB04\SYDEX1 Mounted     0          0          Healthy
DB04\SYDEX2 ServiceDown 0          0          Unknown
```

Despite this, you should not rely on Dynamic Quorum or deliberately cause a situation where a majority of DAG members are offline, as there are still conditions where even Dynamic Quorum can't prevent the DAG from going offline.

## Auto-Reseed in Action

When a disk fails in an Exchange Server 2013 DAG member the Autoreseed workflow begins. However, the following conditions must be met for Autoreseed to take place:

1. The database copies are not blocked from resuming replication or reseeding.
2. The logs and databases files for the database are collocated on the same volume.
3. The logs and database folder structure matches the naming convention required for Autoreseed.
4. There are no other database copies on the volume that are in an "Active" state.
5. All database copies on the volume are in a "FailedAndSuspended" state.
6. The server has no more than 8 "FailedAndSuspended" database copies.

If those conditions are met then Autoreseed can attempt to resolve the issue.

The Autoreseed workflow begins with detection of the failed volume.

1. Database copies are regularly checked to see whether any of them have been at a status of "FailedAndSuspended" for 15 minutes or longer. This is the state that a database copy will be in when there is an underlying storage issue. The 15 minute threshold exists to ensure that remedial action is not taken too quickly.

```
Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 10:19:46 PM
Event ID: 1109
Task Category: Auto Reseed Manager
Level: Information
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager is starting repair workflow 'FailedSuspendedCopyAutoReseed' for
database 'DB01'.

WorkflowLaunchReason: Database copy 'DB01\MELEX1' encountered an error during log replay.
Error: The system cannot find the path specified
```

2. The server attempts to resume the FailedAndSuspended database copy 3 times. An event log entry is logged for each attempt.

```
Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 10:19:46 PM
Event ID: 1124
Task Category: Auto Reseed Manager
Level: Information
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager is beginning attempt number 1 of execution stage 'Resume' for
database copy 'DB01' as part of repair workflow 'FailedSuspendedCopyAutoReseed'.

WorkflowLaunchReason: Database copy 'DB01\MELEX1' encountered an error during log replay.
Error: The system cannot find the path specified
```

After three attempts an event log entry is logged noting that the maximum number of attempts has passed and the workflow will now move on to the next step, which is to attempt to assign a spare volume to replace the failed one.

```
Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 11:04:46 PM
Event ID: 1119
Task Category: Auto Reseed Manager
Level: Error
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager failed to resume database copy 'DB01' as part of repair workflow
'FailedSuspendedCopyAutoReseed' after a maximum of 3 attempts. The workflow will next
attempt to assign a spare volume and reseed the database copy.

WorkflowLaunchReason: The Microsoft Exchange Replication service is unable to create
required directory C:\ExchangeDatabases\DB01\db01.log for DB01\MELEX1. The database copy
status will be set to Failed. Please check the file system permissions. Error:
System.IO.DirectoryNotFoundException: Could not find a part of the path
'C:\ExchangeDatabases\DB01\db01.log'.
```

3. The server attempts to assign a spare volume once per hour for up to 5 attempts. An event log entry is logged for each attempt.

```
Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 11:04:46 PM
Event ID: 1124
Task Category: Auto Reseed Manager
Level: Information
Keywords:
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager is beginning attempt number 1 of execution stage 'AssignSpare' for database copy 'DB01' as part of repair workflow 'FailedSuspendedCopyAutoReseed'.

WorkflowLaunchReason: The Microsoft Exchange Replication service is unable to create required directory C:\ExchangeDatabases\DB01\db01.log for DB01\MELEX1. The database copy status will be set to Failed. Please check the file system permissions. Error:
System.IO.DirectoryNotFoundException: Could not find a part of the path 'C:\ExchangeDatabases\DB01\db01.log'.
```

If the allocation of a spare volume is successful another event log entry is logged to record the outcome. With a spare volume allocated the next part of the workflow can run which attempts to reseed the database copies onto the new volume.

```
Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 11:04:46 PM
Event ID: 1125
Task Category: Auto Reseed Manager
Level: Information
Keywords:
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager has successfully assigned spare volume '\\?\Volume{6e77b6f8-6f83-49f1-ae48-60aa9419cd19}\' mounted at 'C:\ExchangeVolumes\Volume3\' for database copy 'DB01' as part of repair workflow 'FailedSuspendedCopyAutoReseed'. The workflow will next attempt to reseed the database copy.

WorkflowLaunchReason: The Microsoft Exchange Replication service is unable to create required directory C:\ExchangeDatabases\DB01\db01.log for DB01\MELEX1. The database copy status will be set to Failed. Please check the file system permissions. Error:
System.IO.DirectoryNotFoundException: Could not find a part of the path 'C:\ExchangeDatabases\DB01\db01.log'.
```

4. The server attempts to reseed the database copies to the new volume, with up to 5 attempts at 1 hour intervals. Event log entries are logged for each attempt.

```

Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 2/09/2014 11:19:46 PM
Event ID: 1117
Task Category: Auto Reseed Manager
Level: Information
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
Automatic Reseed Manager throttled repair workflow 'FailedSuspendedCopyAutoReseed' for database 'DB01'. Details: The Automatic Reseed Manager encountered an error: The automatic repair operation for database copy 'DB01\melex1' will not be run because it has been throttled by the throttling interval of '01:00:00'.

WorkflowLaunchReason: The Microsoft Exchange Replication service is unable to create required directory C:\ExchangeDatabases\DB01\db01.log for DB01\MELEX1. The database copy status will be set to Failed. Please check the file system permissions. Error: System.IO.DirectoryNotFoundException: Could not find a part of the path 'C:\ExchangeDatabases\DB01\db01.log'.

```

If the reseed operation is successful an event log entry is written for each successful database reseed.

```

Log Name: Microsoft-Exchange-HighAvailability/Seeding
Source: Microsoft-Exchange-HighAvailability
Date: 3/09/2014 12:10:17 AM
Event ID: 826
Task Category: Seeding Target
Level: Information
User: SYSTEM
Computer: MELEX1.exchange2013demo.com
Description:
DB Seeding has completed for the local copy of database 'DB01' (1b3363f6-7f82-41ca-953b-2c295c1896a9).

```

5. If the reseed process was not successful after 5 attempts, it stops trying.
6. After 3 days, if the database copies are still "FailedAndSuspended", the whole workflow begins again from the start.

As you can see Autoreseed is quite intelligent and effective, resolving a straight-forward issue like storage failure with no manual intervention by the administrator.

# Mailbox Server Summary

Exchange Server 2013 packs a lot of features and intelligence into the Mailbox server role that allow it to deliver highly available and site resilient services to the organization.

With so much going on in the Mailbox server role it can be tempting to design complex solutions for business critical Exchange deployments. However in reality the more complex solutions tend to be more costly, more difficult to manage, and in many cases less reliable. A simple design that leverages the capabilities of the DAG such as lagged copy enhancements and Autoreseed will more often lead to a reliable solution for your organization.

# Transport High Availability

Exchange Server 2013 introduces some architectural changes for how Transport works, as well as introducing new Transport features.

Understanding these changes is important to understanding how Transport high availability works overall. Fortunately for most Exchange Server 2013 environments, Transport high availability works quite well on its own without any significant effort by the administrators.

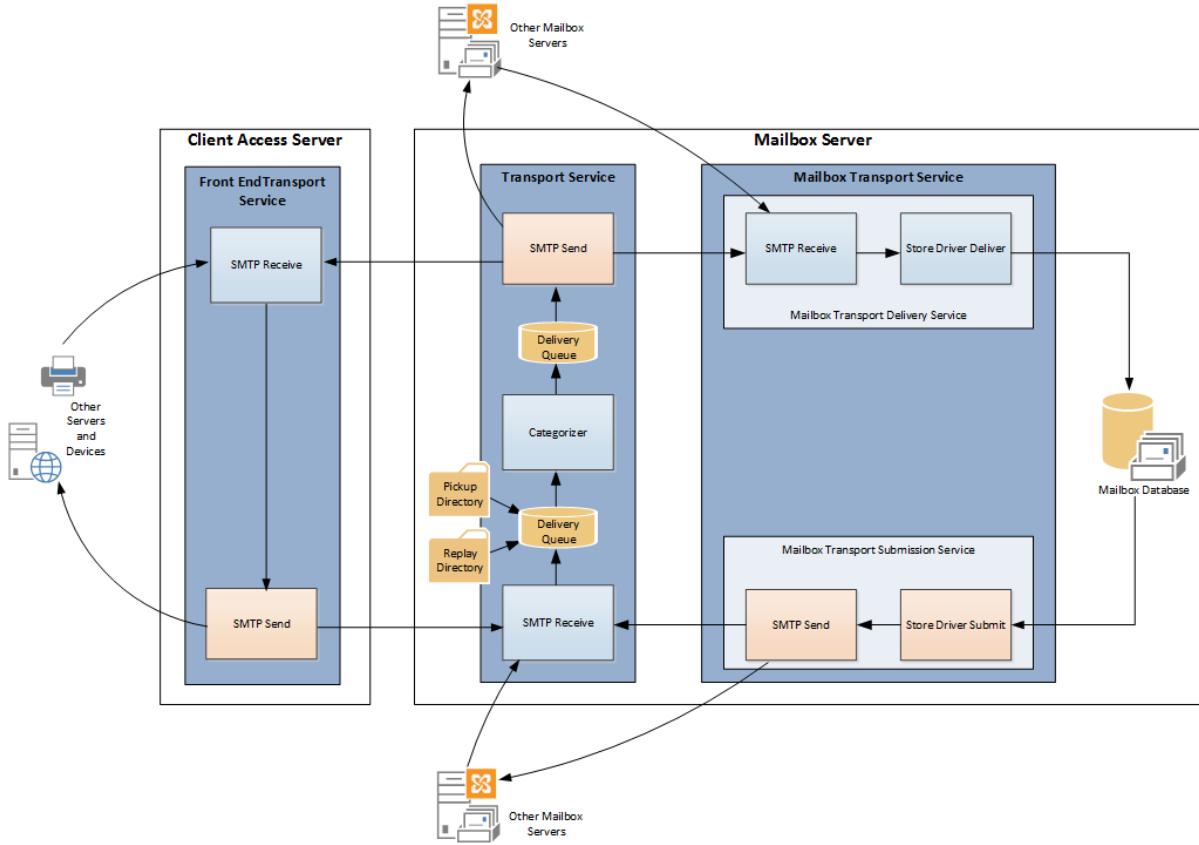
However it is still important to have an understanding over how Transport is working behind the scenes, and what you can do to improve the resilience and availability of Transport, both of which will improve your ability to troubleshoot Transport issues in a high availability environment.

## Exchange Server 2013 Transport Architecture

The Transport architecture of Exchange Server 2013 has changed due to the changes that were made to the server roles. Transport services are now split over the two main server roles; Client Access, and Mailbox.

The Transport pipeline in Exchange Server 2013 is made up of all of the components that work together to deliver emails in the Exchange organization. These components include connectors, services and queues.

In this diagram you can see the complex relationship between these components and how email flows between them.



## Client Access Server Transport Services

Just as the Client Access server is considered a “thin, stateless proxy” for client connectivity protocols such as HTTPS, the same is true for Transport (SMTP) connectivity as well.

The Client Access server role hosts the Front End Transport service. This service is involved in inbound mail flow, as well as optionally being involved in outbound mail flow.

## Mailbox Server Transport Services

The Mailbox server role hosts the Transport service and the Mailbox Transport services.

The Transport service performs a role similar to the Hub Transport server in previous versions of Exchange. The Transport service can perform inspection and queueing of emails, and is responsible for routing email between the Front End Transport service and the Mailbox Transport service. This includes routing email to or from those services on other Exchange servers in the organization.

The Mailbox Transport services are two services that communicate between the Transport service and the mailbox database itself.

The Mailbox Transport Delivery service receives email from the Transport service and delivers it to each of the recipient's mailboxes in their respective databases. When a database availability group has been deployed only the Mailbox Transport Delivery service on the DAG member hosting the active database copy for the target mailbox will handle the delivery of incoming email to the database.

The Mailbox Transport Submission service connects to active mailbox databases on the local server using RPC to retrieve messages. It then submits the messages using SMTP to the Transport service on either the local server or another Mailbox server.

## Routing Destinations

The routing destination for an email message is the final destination that Exchange has determined for the message based on the recipient information. The routing destination for an email message may be one of the following:

- A mailbox database, when the recipient has been determined to be a mailbox within the Exchange organization.
- A connector, when the recipient has been determined to be outside of the organization.
- A distribution group expansion server, when a distribution group has been used that has a specific expansion server configured. After a distribution group has been expanded the message may be duplicated, or "bifurcated", into multiple individual messages with different routing destinations.

## Delivery Groups

A routing destination has one or more Mailbox servers that are responsible for delivering messages to that destination. These are referred to as a delivery group. Depending on the topology of the Exchange organization a message may need to pass through multiple delivery groups to reach the routing destination.

In an Exchange Server 2013 organization a delivery group may be:

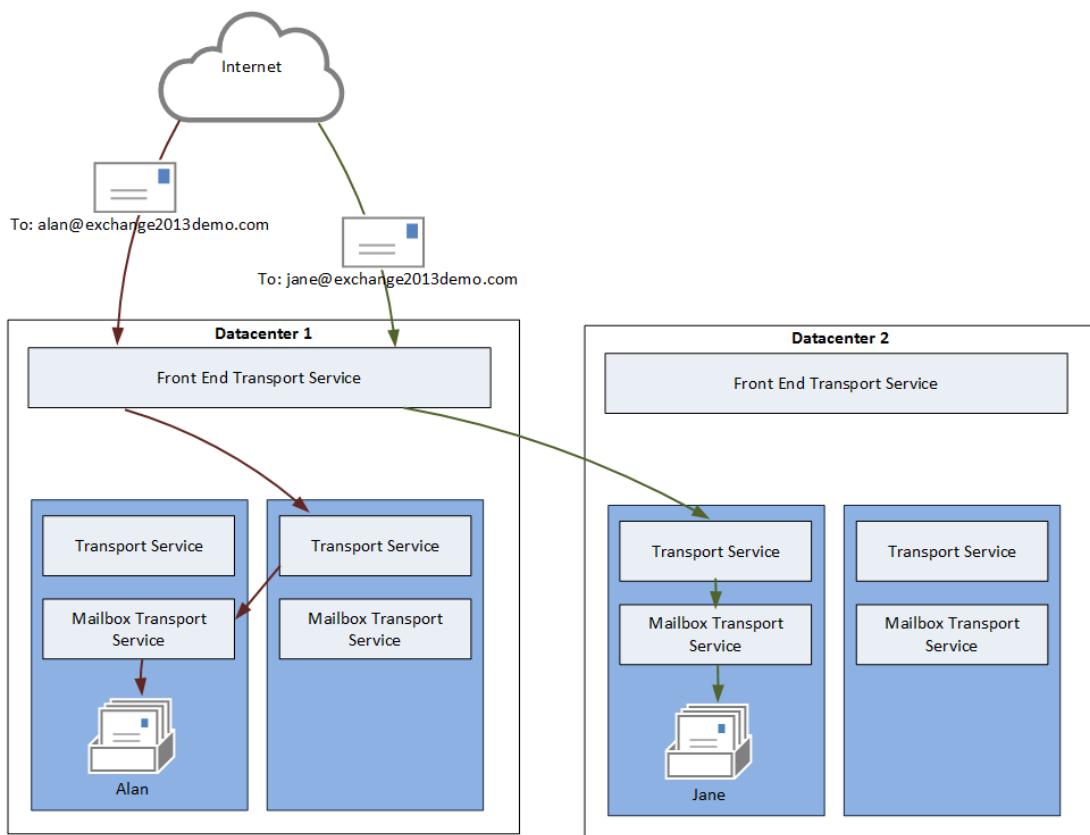
- A database availability group – the members of a DAG are considered to be the delivery group for any mailboxes hosted on databases within that DAG.
- A group of transport services in an Active Directory site – if the mailbox is not hosted in a DAG then the transport servers in the same site as the mailbox are used as the delivery group.
- Connector source transport servers – if a message is being sent over a send connector then the source transport servers configured on that connector are the delivery group.
- An Active Directory site – if a hub site has been configured and provides the least-cost route to the destination then the transport servers in that site are a delivery group.

## Front End Transport Service Email Routing

The Front End Transport service performs no inspection, filtering or queueing of inbound email. Instead it locates the appropriate Transport service on a Mailbox server to route the connection to, which is based on the target delivery group identified by looking up the recipients in Active Directory.

For email messages with a single recipient a Mailbox server in the target delivery group is chosen. If the target delivery group is a DAG that spans multiple Active Directory sites then servers in the nearest AD site are preferred by the Front End Transport service.

Even though the recommended practice is to install multi-role servers it is helpful to visualize mail flow in terms of separate server roles.



For email messages with multiple recipients the first 20 recipients are used to determine the closest target delivery group.

If Front End Transport can't locate a healthy Transport service to route to the connection fails.

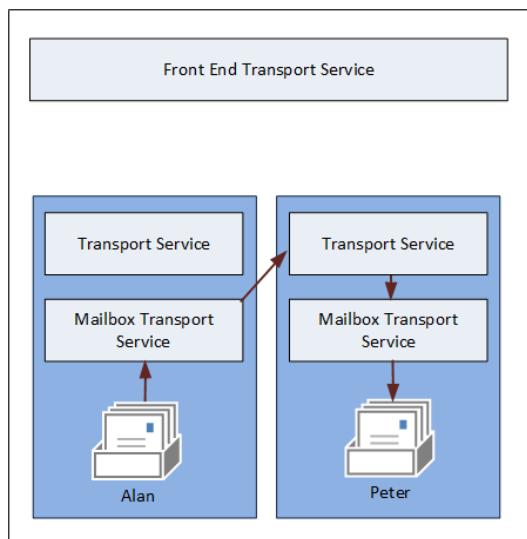
**Real World:** When there are no available Transport services to connect to the sending server will receive a response of "451 4.7.0 Temporary server error. Please try again later. PRX3".

## Mailbox Transport Service Email Routing

For email messages sent to recipients within the Exchange organization the Transport service on Mailbox servers never communicates directly with databases.

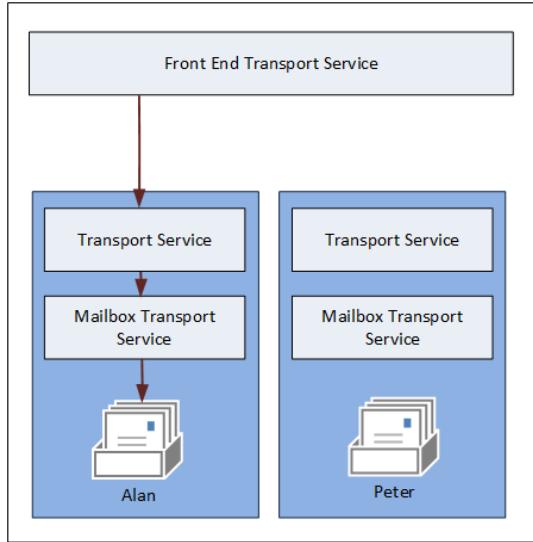
The Mailbox Transport Submission service connects to the mailbox database using RPC, and then transfers the message via SMTP to the Transport service on the local Mailbox server, or on another Mailbox server if necessary.

For example, if an email message is sent by a mailbox in database DB01 that is active on server SYDEX1, and the recipient is a mailbox in database DB02 that is active on server SYDEX2, then the Mailbox Transport Submission service on SYDEX1 will send the message via SMTP to the Transport service on SYDEX2.

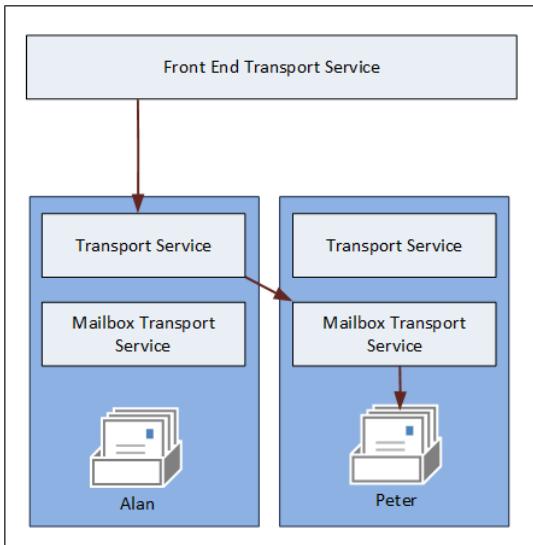


If the Transport service processing an email message is running on a Mailbox server that is already within the target delivery group for the message then incoming email messages are sent via SMTP to the Mailbox Transport Delivery service on the Mailbox server hosting the mailbox database for that recipient. The Mailbox Transport Delivery service then connects to the database to deliver the message to the mailbox.

For example, if the server SYDEX1 is processing a message for a mailbox in database DB01 that is active on SYDEX1, then the Transport service sends the message via SMTP to the Mailbox Transport Delivery service on SYDEX1.



However, if the recipient is a mailbox in database DB02 that is active on SYDEX2, then the message is sent via SMTP to the Mailbox Transport Delivery service on SYDEX2 instead.

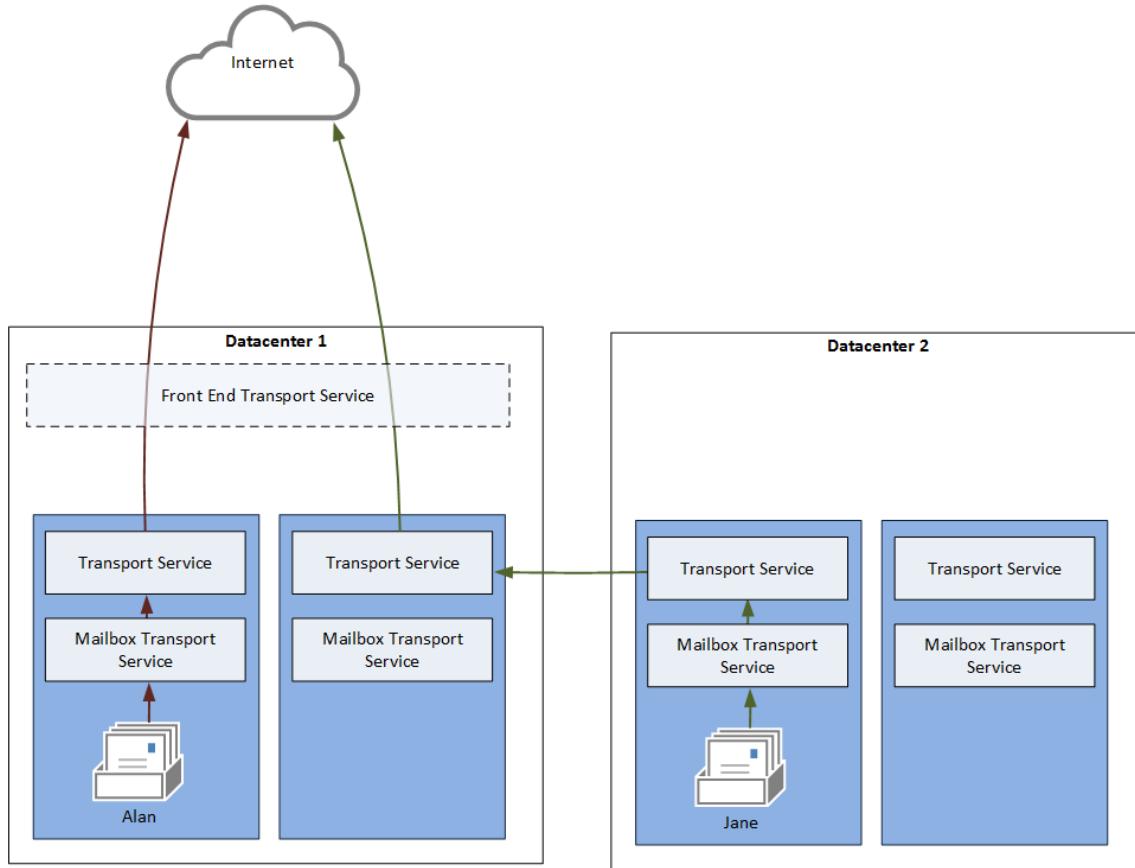


If the Mailbox server is not already within the target delivery group then the email message is sent via SMTP to a Mailbox server in the target delivery group instead.

For email messages sent to recipients outside of the Exchange organization the Transport service sends the message via SMTP to a Mailbox server that is a source Transport server for the send connector for that destination.

If the Mailbox server is itself a source transport server for the send connector then it sends the message via SMTP to the destination configured on the send connector, which may be either a smart host or the IP address of a server determined by looking up MX records in DNS.

Send connectors can optionally be configured to proxy the outbound connection through a Client Access server (i.e., through a Front End Transport Service).



In the case of Office 365 Hybrid, the servers chosen to participate in Hybrid transport will be used to route email messages to mailboxes in Exchange Online by using the “Outbound to Office 365” send connector.

## High Availability for Transport Services

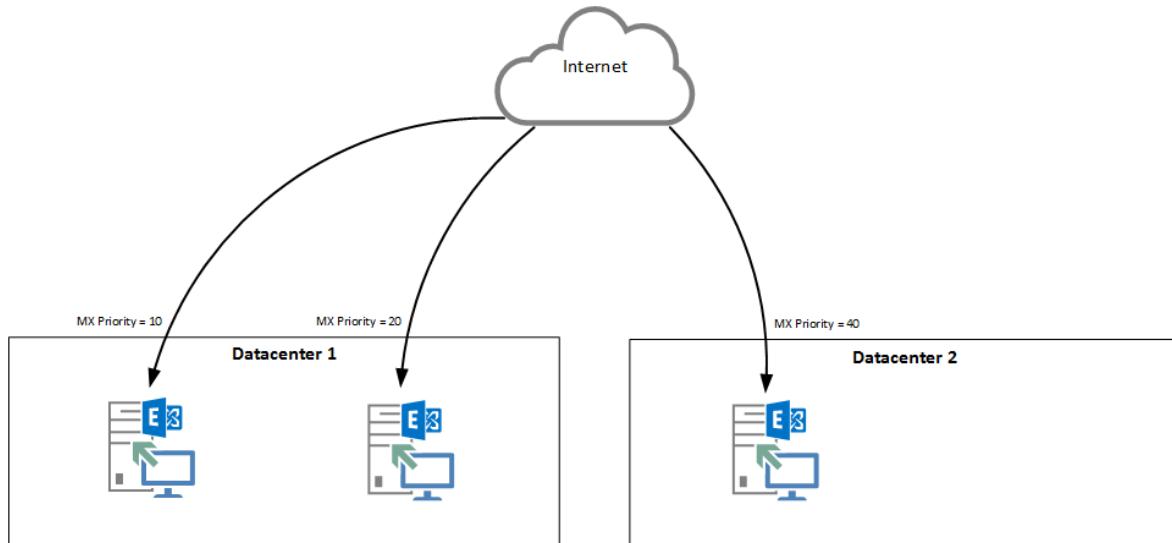
With email routing decisions being made automatically by Exchange Server 2013 you may be wondering where the opportunities are for adding more resilience to the process. Let’s take a look at design and configuration decisions that you can make that impact high availability for transport services.

### Inbound Internet Email

Emailing coming in to the Exchange organization from the internet is received by the Front End Transport service. This is true whether the email comes in directly to Exchange or comes in via a smart host.

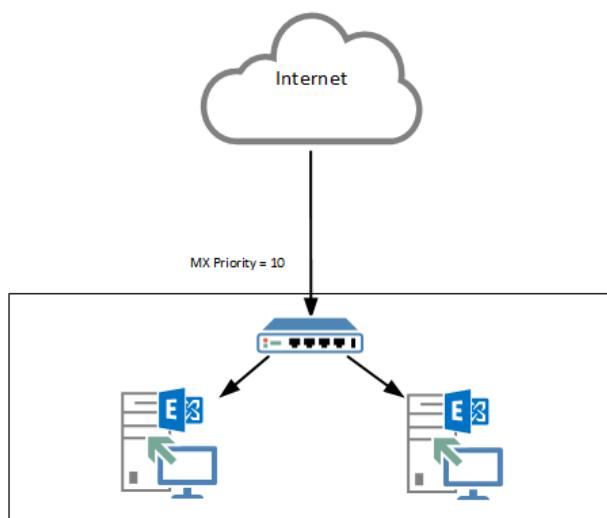
Inbound email routing is determined by MX records in public DNS. These typically resolve to public IP addresses on the organization's edge network device, and are NATed to smart hosts or Exchange servers in the corporate network.

Using multiple MX records in DNS allows you to specify multiple inbound routes for email. These can be in the same or separate datacenters, depending on the level of resilience your organization needs.



**Note:** For more information about MX records read [Email Fundamentals: What is an MX Record](#)<sup>12</sup>.

When only a single public IP address is available you can still achieve a level of high availability for inbound email by using a load balancer.



<sup>12</sup> <http://exchangeserverpro.com/mx-record/>

Neither of these options requires any special configuration on the Exchange servers themselves, because the Client Access server is automatically configured with a receive connector that is capable of receiving inbound internet email.

However, an important consideration is whether the load balancer passes the real source IP address of the sending server to the Exchange server, which is crucial for access control, message hygiene, and accurate logging.

## Outbound Internet Email

Email sent from the Exchange organization to the internet is routed via send connectors. Outbound email may be sent to a smart host or directly to the MX record of the destination email domain.

Send connectors can be configured with multiple source transport servers to provide high availability for the connector.

new send connector

A send connector sends mail from a list of servers with transport roles or Edge Subscriptions. [Learn more...](#)

\*Source server:  
Associate this connector with the following servers containing transport roles. You can also add Edge Subscriptions to this list.

SERVER	SITE	ROLE
SYDEX1	exchange2013demo.com/Configuration/Sites/Sydn...	Mailbox, Clien...
SYDEX2	exchange2013demo.com/Configuration/Sites/Sydn...	Mailbox, Clien...

Multiple send connectors can also be created. For example, an Exchange organization with two geographic regions may be configured with a send connector for each region so that outbound email can take the most efficient route to the internet.

**Note:** For more information about how to create and configure send connectors read [Configuring Outbound Mail Flow in Exchange Server 2013](#)<sup>13</sup>.

<sup>13</sup> <http://exchangeserverpro.com/configuring-outbound-mail-flow-in-exchange-server-2013/>

## Internal SMTP Senders

Within an Exchange environment it is common for other applications or devices to need an SMTP service available for sending email messages.

The Front End Transport service is already configured to permit any sender to send email to recipients within the Exchange organization, so there is no special configuration required for that scenario.

However some applications or devices may need the capability to send email to external recipients. If the applications or devices are able to make authenticated SMTP connections to Exchange then once again there is no special configuration required for Exchange itself.

For situations where authenticated SMTP is not possible then Exchange Server 2013 can be configured with additional receive connectors that can handle this anonymous SMTP relay requirement.

**Note:** For more information on how to create and configure these receive connectors read [How to Configure a Relay Connector in Exchange Server 2013<sup>14</sup>](#).

To provide high availability for internal SMTP senders a load balancer can be used to distribute the traffic across the available Front End Transport services running on Client Access servers.

Although the use of DNS round robin is a valid high availability solution for other Client Access protocols such as HTTPS, it is generally not suitable for SMTP because many applications and devices are not intelligent enough to automatically try multiple connections to different IP address returned by DNS, and will instead simply consider the SMTP connection failed if the first IP address does not respond.

**Warning:** When a source NAT is used with a load balancer the Exchange server loses visibility of the real source IP address of the application or device sending the email. All SMTP connections will appear to be coming from the load balancer's IP address. This makes it impossible to control who can relay emails via the Exchange server based on IP addresses.

If you are planning to load balance SMTP in this manner use a load balancer configuration that passes the real IP address of the source to the Exchange server, or apply access control on the load balancer itself.

<sup>14</sup> <http://exchangeserverpro.com/exchange-2013-configure-smtp-relay-connector/>

# High Availability Features of Mailbox Transport Services

In addition to the high availability design and configuration decisions that Exchange administrators can make there are also some features built in to the transport services on Mailbox servers that improve the availability and resilience of email in transit, as well as email that has already been delivered to mailboxes.

## Shadow Redundancy

Shadow redundancy first appeared in Exchange Server 2010. Shadow redundancy keeps email messages in the transport database until the next hop in the delivery path verifies successful delivery of the message. If the verification is not received then the message still exists in the transport database and can be resubmitted.

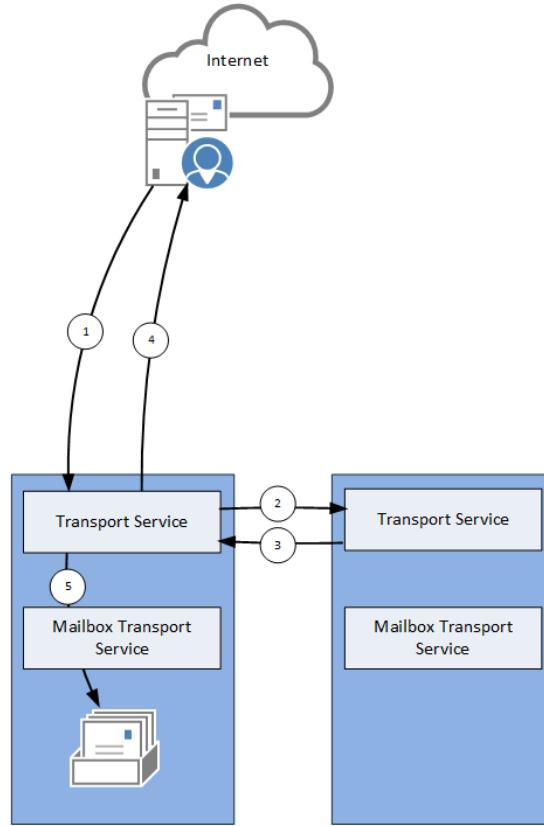
A weakness of the Exchange 2010 implementation of shadow redundancy was in the handling of messages received by Exchange 2010 transport servers sent by source servers that do not support shadow redundancy, such as a third party email server or smart host. In this scenario the Exchange 2010 transport service would delay acknowledgement of the receipt of the message so that if delivery to the next hop failed the original server will not yet have received an acknowledgement for the original delivery attempt, so it will be able to resubmit its copy of the message.

However, if the maximum acknowledgement delay (30 seconds by default) had lapsed before the message could be sent on to the next hop, for example due to an internal delivery delay, then the Exchange 2010 server would send an acknowledgement to the sending server anyway. This meant that redundancy was not guaranteed in this scenario and was considered best effort only.

Exchange Server 2013 improves shadow redundancy by always making a redundant copy on another transport server for any message received from a source server that does not support shadow redundancy.

When an incoming email message arrives from an SMTP server

1. The Transport service on the Exchange server in the organization receives the message. Remember, the Front End Transport service is just a thin, stateless proxy in this scenario.
2. Before acknowledging successful receipt of the message, the first Exchange server will send a copy of the message to the Transport service on another Mailbox server.
3. The second Transport service stores the copy of the message in a shadow queue, and sends an acknowledgement to the first server.
4. The first Exchange server can now send an acknowledgement to the originating SMTP server, and the SMTP session can be closed.
5. The message is delivered to the recipient.



Shadow redundancy is enabled by default for an Exchange Server 2013 organization and should be left enabled.

Shadow redundancy is performed by the Transport service on Mailbox servers, as this is the only service in the transport pipeline that does any queuing or caching of email messages. Obviously shadow redundancy requires more than one Mailbox server, but those servers do not need to be members of a DAG.

However, whether the servers are members of a DAG or not does influence some of the behavior for shadow redundancy.

If the Mailbox server processing an email message is not a DAG member, then the Mailbox server it selects to host the shadow copy of the message must be in the same Active Directory site.

On the other hand, if the Mailbox server is a DAG member then the server chosen for the shadow copy must be a member of the same DAG. For DAGs that span multiple sites a DAG member in a different AD site is chosen if one is available, which provides even more resilience for shadow redundancy.

**Real World:** When you are designing a Database Availability Group to span multiple physical datacentre locations it is recommended that each datacentre be a separate Active Directory Site so that shadow redundancy can also be site resilient.

## Safety Net

Safety Net is the evolution of the transport dumpster from Exchange Server 2007 and 2010. The transport dumpster held copies of email messages that had been successfully delivered to mailboxes so that they could be resubmitted if a lossy failover occurred.

This meant that the transport dumpster was only available when Exchange 2007 CCR clusters or Exchange 2010 DAGs were used. It was not available for standalone Mailbox server scenarios.

Exchange Server 2013 improves on the capabilities of transport dumpster and the feature is now called Safety Net.

Safety Net stores copies of successfully delivered messages in the transport database for a configurable period of time. The default value of this is 2 days.

```
[PS] C:\>Get-TransportConfig | Select SafetyNetHoldTime  
SafetyNetHoldTime  
-----  
2.00:00:00
```

The value can be changed, and applies to the entire Exchange organization. You can't set different Safety Net hold times for individual servers in the organization.

```
[PS] C:\>Set-TransportConfig -SafetyNetHoldTime 7.00:00:00  
WARNING: Setting 'SafetyNetHoldTime' to a value lower than 'ReplayLagTime' in a Mailbox database copy  
can cause irrecoverable data loss. Please ensure that the 'SafetyNetHoldTime' parameter is set to a  
value equal or greater than the 'ReplayLagTime' parameter, which is set using the Set-  
MailboxDatabaseCopy cmdlet.
```

Messages are resubmitted by Safety Net automatically in two different scenarios:

- After a lossy failover of a mailbox database in a DAG
- During activation of a lagged mailbox database copy

For more information on lagged database copies refer to the chapter on Mailbox server high availability.

**Real World:** It is recommended to set the Safety Net hold time to match the replay lag time of lagged database copies. However, you need to take into account the increased disk storage requirements that this will create for the Transport database on Mailbox servers, which will need to store all of that Safety Net data.

## How Safety Net and Shadow Redundancy Work Together

It is said that Safety Net begins where shadow redundancy ends. In other words, shadow redundancy is responsible for guaranteeing delivery of email messages in transit, whereas Safety Net is responsible for guaranteeing resubmission of previously delivered messages during mailbox database failure and recovery scenarios.

However the two features also work together in some ways. Safety Net is itself made redundant by Shadow Safety Net. In a failure scenario where the Primary Safety Net is unavailable the resubmission requests for email messages can be serviced by the Shadow Safety Net instead.

In addition, when DAGs are used shadow redundancy does not need to maintain copies of messages until they have been replicated to every database copy in the DAG. Instead the DAG will use Safety Net to recover from scenarios where a failure occurs before the delivered message had fully replicated.

**Note:** Resubmission of messages seems like a recipe for duplicate items to occur in end user mailboxes. However, Exchange Server 2013 is intelligent enough to avoid duplicate message redelivery in most cases.

## Preserving Safety Net and Shadow Redundancy

Except for the configuration of the Safety Net hold time most of the operations of Safety Net and shadow redundancy happen automatically. Optimising the behaviours of Safety Net and shadow redundancy mostly comes down to correct server role replacement, for example ensuring that multi-site DAGs span multiple Active Directory Sites rather than one stretched AD Site.

However you can easily undermine Safety Net and shadow redundancy through server maintenance or recovery activities. For example, if an Exchange server is running low on disk space an administrator may identify the transport database as being too large and decide to stop the transport services, remove the large database file, then start the services again so that a new, empty transport database file is created.

The administrator has solved their low disk space problem but in doing so has deleted the database that contains the Safety Net and shadow redundancy copies of email messages for the organization. If they were to repeat this action on multiple servers you can see how they would easily wipe out all of the redundant copies of email messages, undermining future recovery actions.

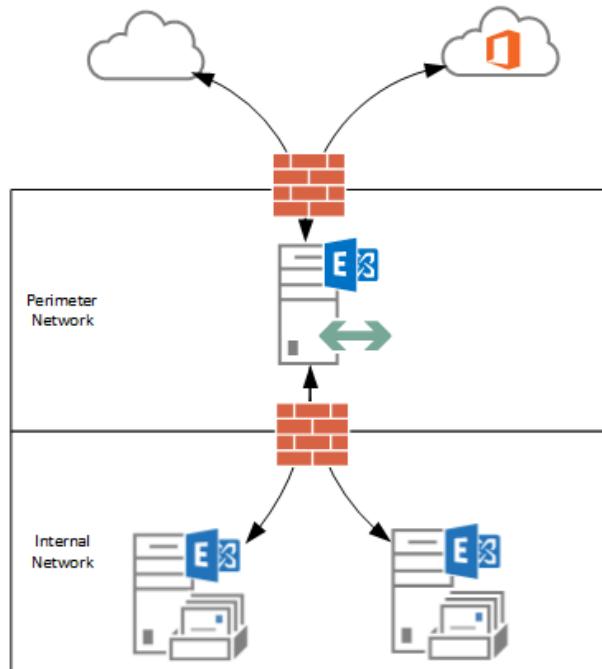
**Real World:** The disk storage requirements for Transport services are often underestimated by Exchange customers. You should always ensure that enough disk capacity exists for the volume that is hosting the transport database files, and also ensure that the volume itself (which is often just the System volume) is protected from hardware failure by using RAID.

# Edge Transport Server High Availability

The Edge Transport role is involved in SMTP communications (email transport) in and out of an Exchange organization.

One or more Edge Transport servers are typically placed in a perimeter network to satisfy the needs of organizations who require no direct connectivity between the internal network and the internet, and prefer to use a Microsoft solution for this rather than a third party product.

Edge Transport can also serve this role for hybrid deployments with Office 365, so that mail flow between the on-premises organization and the cloud passes through the Edge server when centralized mail transport is enabled for the Hybrid configuration.



High availability for Edge Transport servers depends on the network environment of the organization. For example, an organization may have a single datacenter and internet connection, but still want high availability for the Edge Transport role itself even though the datacenter or internet connection are potentially single points of failure.

In comparison, an organization may have multiple datacenters and internet connections, and want both high availability and site resilience.

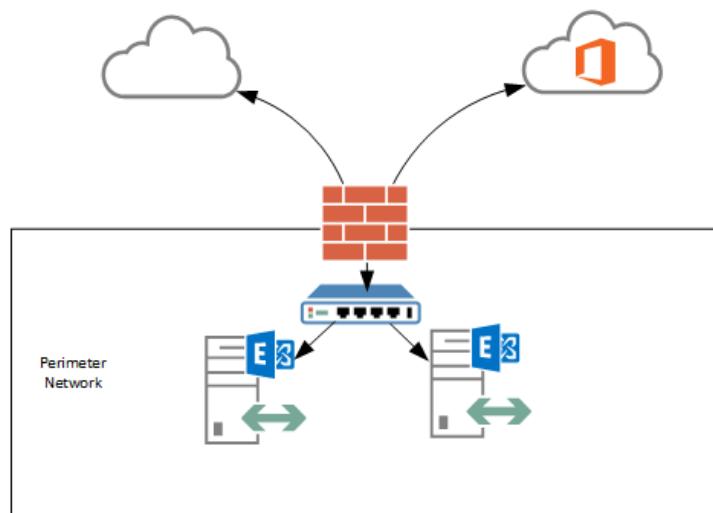
Let's explore some of these scenarios and how the high availability objective for Edge Transport can be achieved.

## Single Datacenter and Internet Connection

In a single datacenter with a single internet connection there are two possibilities for how Edge Transport high availability can be achieved.

Inbound email routing is determined by MX records in public DNS. These typically resolve to public IP addresses on the organization's edge network device that are either assigned directly to the Edge Transport servers or are NATed to the Edge Transport servers.

If an organization has only one public IP address then logically only one Edge Transport server can be NATed for incoming SMTP connections. In this scenario a load balancer can be used to distribute the incoming SMTP traffic between multiple Edge Transport servers, providing high availability for the inbound email route.



However, if multiple public IP addresses exist then the load balancer is not required, as each Edge Transport server can be NATed separately. Inbound email traffic can still be load balanced by the use of multiple MX records with equal priority, and it removes the complexity of a load balancer from the solution which can help avoid situations where the load balancer is directing traffic to an Edge Transport server that has a health problem that the load balancer can't detect with a standard SMTP probe.

When a single Edge Transport server is subscribed to an Active Directory site, two send connectors are created automatically for inbound and outbound mail flow through the Edge server.

On the Edge Transport server EDGE1 an Edge Subscription file is created.

```
[PS] C:\>New-EdgeSubscription -FileName C:\Admin\edge1.xml
```

The file is then copied to a Mailbox server in the Sydney datacenter and used to create the Edge Subscription.

```
[PS] C:\>New-EdgeSubscription -FileData ([byte[]]$([Get-Content -Path "C:\Admin\Edge1.xml" -Encoding Byte -ReadCount 0)) -Site "Sydney"
```

The send connectors can then be seen with the EDGE1 server as the source transport server.

```
[PS] C:\>Get-SendConnector | select identity,sourcetransportservers
```

Identity	SourceTransportServers
-----	-----
EdgeSync - Sydney to Internet {EDGE1}	
EdgeSync - Inbound to Sydney {EDGE1}	

When additional Edge Transport servers are subscribed to the same site they are added as source transport servers for the same connectors.

```
[PS] C:\>New-EdgeSubscription -FileData ([byte[]]$([Get-Content -Path "C:\Admin\Edge2.xml" -Encoding Byte -ReadCount 0)) -Site "Sydney"
```

```
[PS] C:\>Get-SendConnector | select identity,sourcetransportservers
```

Identity	SourceTransportServers
-----	-----
EdgeSync - Sydney to Internet {EDGE2, EDGE1}	
EdgeSync - Inbound to Sydney {EDGE2, EDGE1}	

This has the effect of load balancing the outbound email traffic to the internet across both Edge Transport servers. Email from the internal Exchange servers can traverse either Edge Transport server on its way out to the internet.

If one Edge Transport server is unavailable the internal Exchange servers will continue to use the other Edge Transport server with no interruption to mail flow.

**Real World:** There are still failure scenarios that can interrupt outbound email. For example, if the firewall is not permitting one of the Edge Transport servers to send outbound email then it will queue on that server instead. So even with multiple Edge Transport servers you still need to monitor them for any issues like this that may arise.

## Multiple Datacenters and Internet Connections

When an organization has multiple datacenters and internet connections they can subscribe Edge Transport servers to each site to provide a greater degree of high availability and site resilience than is a single datacenter is capable of.

In this example there are two datacenters, Sydney and Melbourne, each with their own internet connection available for routing email in and out of the organization. An Edge Transport server has been deployed in each datacenter.

As with the single datacenter example, on the Edge Transport server EDGE1 an Edge Subscription file is created.

```
[PS] C:\>New-EdgeSubscription -FileName C:\Admin\edge1.xml
```

The file is then copied to a Mailbox server in the Sydney datacenter and used to create the Edge Subscription.

```
[PS] C:\>New-EdgeSubscription -FileData ([byte[]]$([Get-Content -Path "C:\Admin\Edge1.xml" -Encoding Byte -ReadCount 0)) -Site "Sydney"
```

The send connectors can then be seen with the EDGE1 server as the source transport server.

```
[PS] C:\>Get-SendConnector | select identity,sourcetransportservers
Identity           SourcetransportServers
-----
EdgeSync - Sydney to Internet {EDGE1}
EdgeSync - Inbound to Sydney   {EDGE1}
```

The second Edge Transport server, EDGE2, is subscribed to the Melbourne site instead.

```
[PS] C:\>New-EdgeSubscription -FileData ([byte[]]$([Get-Content -Path "C:\Admin\Edge2.xml" -Encoding Byte -ReadCount 0)) -Site "Melbourne"
```

Now there are additional send connectors configured for that datacenter, using EDGE2 as the source transport server.

```
[PS] C:\>Get-SendConnector | select identity,sourcetransportservers,addressspaces
```

Identity	SourceTransportServers	AddressSpaces
EdgeSync - Sydney to Internet	{EDGE1}	{smtp: *; 100}
EdgeSync - Inbound to Sydney	{EDGE1}	{smtp: --; 100}
EdgeSync - Melbourne to Internet	{EDGE2}	{smtp: *; 100}
EdgeSync - Inbound to Melbourne	{EDGE2}	{smtp: --; 100}

This configuration creates two inbound and outbound routes for email to flow via the Edge Transport servers in each datacenter.

For inbound email flow of course there must also be an MX record that resolves to the public IP address of EDGE2 in the Melbourne datacenter.

The resilience of this configuration can be improved even further by deploying two or more Edge Transport servers in each datacenter and using multiple public IP addresses to point an MX record to each Edge Transport server, or use load balancers to distribute the traffic between Edge Transport servers.

When multiple MX records are being used to route inbound mail flow through multiple datacenters you should consider the impact that this will have on cross-site traffic within your organization.

The sending SMTP server has no awareness of where mailboxes are located within your organization. It simply looks up the MX records for your domain name, chooses the appropriate one to try first, and then sends the email message.

This could result in an incoming email message being delivered to a site that is not the site where the mailbox is actually hosted at the time. Exchange will simply route the email internally to the correct site, so the delivery itself is not likely to be a concern. But you should be aware that this will increase the amount of network traffic that is utilizing the links between the two sites.

**Real World:** There are many different ways for multiple datacentre deployments to be configured. Some organizations may choose to deploy two Edge Transports in their primary datacentre and only one Edge Transport in their secondary datacentre. Others may be forced to use a load balancer in one datacentre due to lack of public IP addresses, but can use multiple IPs in the other. This all comes down to what level of high availability and site resilience you're trying to achieve, and the resources available for you to design your deployment.

# Effect of Connector Costs on Email Routing

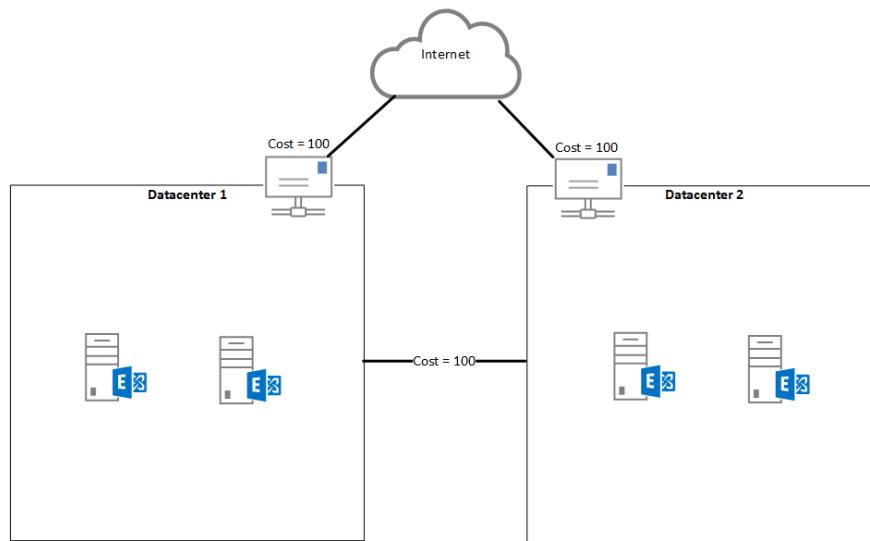
In the multiple datacenter example you may have noticed the values for the AddressSpaces attribute of the send connectors. Although in this example the send connectors have been created by the Edge subscription, this AddressSpaces value also exists on send connectors in organizations without Edge Transport servers.

[PS] C:\>Get-SendConnector   select identity,sourcetransportservers,addressspaces		
Identity	SourceTransportServers	AddressSpaces
EdgeSync - Sydney to Internet	{EDGE1}	{smtp:*;100}
EdgeSync - Inbound to Sydney	{EDGE1}	{smtp:--;100}
EdgeSync - Melbourne to Internet	{EDGE2}	{smtp:*;100}
EdgeSync - Inbound to Melbourne	{EDGE2}	{smtp:--;100}

The value “100” is the cost of the connector. Cost is an arbitrary value assigned to send connectors that is used to determine the relative priority or weighting of each connector.

In other words, a send connector with a cost of “10” will be attempted to be used first instead of another connector with a cost of “50” or “100”.

Active Directory site links also have costs configured on them. For example, in this topology each send connector has a cost of “100” and the AD site link also has a cost of “100”.



When an email is sent by a mailbox located in an active database copy in Melbourne, it calculates the least cost route to the destination. Assuming all routes are available, the least cost route will be directly out via the send connector for Melbourne, which routes via the Edge Transport server EDGE2.

However, if the send connector in Sydney was decreased to a cost of “20”, and AD site link cost was decreased to a cost of “50”, the least cost route from Melbourne to the internet is now via Sydney datacenter.

```
[PS] C:\>Set-SendConnector "EdgeSync - Sydney to Internet" -AddressSpaces {smtp: *;20}
[PS] C:\>Get-SendConnector
Identity                               AddressSpaces          Enabled
-----
EdgeSync - Sydney to Internet        {smtp: *;20}        True
EdgeSync - Inbound to Sydney         {smtp: --;100}      True
EdgeSync - Melbourne to Internet     {smtp: *;100}       True
EdgeSync - Inbound to Melbourne      {smtp: --;100}      True
```

The cost configured on an AD site link can be seen by running Get-ADSiteLink. By default the ExchangeCost is null, and Exchange uses the value assigned for Active Directory cost instead.

```
[PS] C:\>Get-ADSiteLink | fl name,*cost*
Name      : DEFAULTIPSITELINK
Cost      : 100
ADCost   : 100
ExchangeCost :
```

If the AD cost is not suitable for Exchange routing purposes then the ExchangeCost can be configured using Set-ADSiteLink.

```
[PS] C:\>Set-ADSiteLink DEFAULTIPSITELINK -ExchangeCost 50
```

**Real World:** Configuring site link and send connector costs so that outbound email flow goes out a single datacentre can simplify troubleshooting scenarios, but there are some drawbacks as well.

For example, the IP addresses of the secondary datacentre do not build up any positive sender reputation in terms of spam filtering. When your outbound mail flow suddenly occurs from a different public IP address it may cause your organization’s email to be quarantined or blocked by other servers.

If your internet bandwidth costs allow for it, it can be preferable to keep the public IP addresses of both datacenters “warm” and building sender reputation by constantly being in use for outbound email. However, this issue can be avoided by registering SPF records for your domains as well, which indicate to receiving servers which IP addresses are trusted to send for your domain.

# Transport Summary

In this chapter we've covered high availability for Transport in Exchange Server 2013.

The architectural changes in Exchange Server 2013 mean that Transport as a whole encompasses a variety of services on each of the server roles; Client Access, Mailbox, and Edge Transport.

Understanding that new architecture is a critical foundation to deploying and managing Transport high availability.

High availability for Transport requires multiple servers to be deployed, MX records to be published, connectors configured, and a small number of configuration changes such as the Safety Net hold time.

Just as important is the design of other elements of the overall Active Directory and Exchange Server 2013 environment, such as ensuring the correct AD Site boundaries are defined for a multi-site DAG so that shadow redundancy can operate in a site resilient manner.

Finally, viewing Transport as a complete end to end solution rather than just a single component will allow you to avoid undermining high availability with single points of failure.

# High Availability for Unified Messaging

Unified Messaging is a core feature built into Exchange Server 2013 that provides voice access to the messaging environment via an organization's corporate phone system. With modern IP-based Private Branch Exchanges (PBXs) this connection uses a Session Initiator Protocol (SIP) trunk to route voice traffic from the phone system to Exchange. Older analogue based PBX systems can also interact with Unified Messaging via a supported gateway.

The voice access features include basic functionality including Voicemail, as well as more advanced functionality including transcription of voice messages, voice access to email, calendaring and the global address list. Organizations can also take advantage of auto-attendant functionality allowing callers to navigate menus over the phone and get routed to an appropriate telephone extension.

First introduced in Exchange 2007, Unified Messaging began as a separate server role complementing Client Access, Hub Transport, Mailbox and Edge Roles. The Unified Messaging role would typically be installed as a standalone role, especially in HA environments with multiple UM servers installed per Exchange organization to provide failover.

Exchange 2010 improved upon the original functionality delivered with Exchange 2010 by adding auto attendant, voice access to mailboxes and the core framework used today for defining Unified Messaging objects in Active Directory. However the core implementation of Unified Messaging took a similar form with typical deployments using standalone Unified Messaging servers acting as central points for integration with the phone system.

In Exchange 2013 the *Every Server is an Island* concept applies to Unified Messaging as with other roles. Each server hosting the Mailbox role runs Unified Messaging, providing services for users with mailboxes on active databases.

This evolution helps ensure that smaller organizations can implement a highly available UM platform without excessive hardware, and larger organizations automatically scale their UM high availability. The simplification of the HA infrastructure does have some new challenges which we'll cover within this chapter.

## Unified Messaging Concepts

It is important to understand the fundamentals of Unified Messaging before deployment. Although our focus is on high availability we'll cover core concepts of Unified Messaging so that you are armed with the pre-requisite knowledge necessary to understand the purpose of components and relevant configuration.

### UM Dial Plans

A Unified Messaging Dial Plan borrows its name and purpose from the world of the phone system that the UM role bridges. On a traditional phone system, a dial plan relates to the configuration used to define the block of numbers allocated to users. For example an organization may choose to provide a four-digit extension starting with the number 4 to users within a specific office. This is generally specified by using a *mask*, where the character X is used as a wildcard – e.g. 4XXX

We configure dial plans within Exchange to match with the corresponding dial plans in the IP phone system. In general these are a one to one mapping, but in certain circumstances, such as if many dial plans are configured (for example multiple IP-PBX dial plans covering number masks from 40XX to 49XX could be combined into a single dial plan within Exchange using a number mask of 4XXX).

### UM Mailbox Policies

Configuration for users can be specified in two places. The defaults for a user are contained within a UM Mailbox Policy that is attached to a UM Dial Plan. Settings include policies for PIN settings, such as how many digits are required, or whether the user can use text to speech. Multiple UM Mailbox policies can be created to ensure the needs of groups of users within a dial plan can be satisfied. On a per-user basis many of these settings can be overridden. UM Mailbox Policies work in much the same way as other user policies within Exchange, like Retention Policies or ActiveSync Policies. They are similar to a group in use and a user can only be a member of one policy.

### IP Gateways

Within the context of Unified Messaging, an IP gateway is simply a logical construct that defines the IP address or DNS name of the IP-PBX gateway that will attempt to contact Exchange. This allows us to ensure only trusted hosts can communicate with the Unified Messaging servers and for a multi-site deployment a number of IP Gateways are likely to be defined, each relating to the local IP-PBX that will attempt to initiate communications with Exchange.

## UM Hunt Groups

In the world of the IP-PBX a Hunt Group usually relates to a single number assigned to a group of people, such as an inbound sales number that will “hunt” for a free person to answer by ringing multiple extensions.

In Exchange Unified Messaging the UM Hunt Group serves a similar purpose but relates to a single number being assigned to multiple Exchange Servers allowed to accept a call within the relevant dial plan. It’s basically a way of grouping Exchange Servers together within a dial plan.

## UM Auto Attendants

If you have called your bank or utility provider you have probably encountered an automated attendant. Auto attendants allow a switchboard operator to be replaced with a menu that the caller can navigate via voice or DTMF input. UM Auto Attendants are defined as objects in Exchange, and then the prompts and menu customized by either an admin or user within the organization given appropriate permissions. Multiple auto attendants can be created and chained together, providing a tree menu to callers. As well as being used to direct callers to a specific UM-enabled mailbox to leave a message, auto attendants can route callers to extensions on the PBX and provide access to the Global Address List.

## Connecting to Phone Systems

Microsoft Exchange uses the Session Initiator Protocol (SIP) to communicate with Private Branch eXchange (PBX) systems. Although just ten years ago the world was only just beginning to move away from analogue phone systems, many phone systems today are IP-based, and most of those also use the SIP protocol to communicate. In general Microsoft support many common vendors, such as its own Lync system, Avaya, Cisco and others, the connecting IP-PBX often needs to be a minimum software version.

For older systems that do not support SIP, or IP-based systems that are not supported with Unified Messaging a gateway is usually required. These either convert the analogue format to and from SIP and route it to Exchange or act as a bridge between the unsupported IP-PBX, correcting anything unusual it does, or indeed anything unusual Exchange does.

Under the hood. SIP isn't overly complex for an administrator familiar with SMTP or HTTP protocols. Because SIP is concerned with the initiation of communications rather than the audio stream itself, it is a text based protocol. A typical SIP session will negotiate who and where, and then negotiate the audio protocol before handing over to a binary audio stream. This makes debugging SIP communications similar in process to debugging SMTP mail flow because the conversation can be replayed via a text file, which will often show the issue in front of a careful eye.

# Unified Messaging Components

In Exchange 2013 components of Unified Messaging are installed onto both the Client Access and Mailbox server roles – and of course multi-role servers. The way this is implemented represents a big change to the way UM works compared to previous versions. The Client Access role now hosts the UM Call Router Service, and the Mailbox role hosts the UM Service.

## The UM Call Router Service

If you are familiar with the Client Access role then you will know that the primary purpose it serves is to either proxy or redirect traffic. It isn't the source of communications within Exchange and it doesn't directly access mailboxes and respond to users. For many protocols accessing Exchange, like SMTP or HTTPS it acts as a protocol-aware proxy server. With UM the behaviour is to use the SIP REDIRECT verb to direct the IP gateway to speak to the appropriate mailbox server.

By default the UM Call Router service listens on the following ports:

Protocol	Port	Purpose
TCP	5060	SIP unsecured
TCP	5061	SIP secured

## The UM Service

This service takes on the traditional role assumed by the UM role – after traffic is redirected from the UM Call Router Service, a normal SIP communication begins and after negotiating the audio stream parameters via the Session Description Protocol it uses the Real-Time Protocol (RTP) to pass audio back and forth. The UM service uses a number of worker processes to service requests and provide scalability, therefore it requires a few more known ports along with ports open for audio:

Protocol	Port	Purpose
TCP	5062	SIP unsecured to UM service
TCP	5063	SIP secured to UM service
TCP	5065 and 5067	SIP unsecured to UM worker processes
TCP	5066 and 5068	SIP secured to UM worker processes
TCP	1000 – 65535 (negotiated during SIP session)	RTP audio steam port

# Certificate Requirements

If you are using a modern system like Microsoft Lync as your phone system then it is very likely you will want to use SIP secured for communication, which simply means that traffic is encrypted using the same TLS-based encryption used with HTTPS and SMTP.

That means during the design of a highly available infrastructure we will need to ensure that we take into account certificate requirements during the specification of SSL certificates, as for UM to function properly it will expect the fully qualified domain name (FQDN) of each Exchange Server specified.

This can be accomplished by either:

- Adding each Exchange Server FQDN to the SAN (Subject Alternative Name) SSL certificate used on all Exchange Servers. If you've got a large number of Exchange Servers you may prefer not to do this.
- Creating a new SSL certificate for each Exchange Server that will only be assigned to either UM or UM call routers. This must be a certificate the PBX server trusts, so should be signed by either an internal or third party certificate authority.

Most organizations will opt for the latter option as the main third-party certificate won't benefit from using the FQDN of each server for any other service, and some organizations might consider exposing the server names as a security risk. Additionally, most organizations deploying Lync will use an internal certificate authority for internal Lync servers and therefore have a trusted internal CA available.

With any option, consider the ongoing maintenance of certificates. Certificates must be renewed before they expire, as if they do not they cause an outage. If all certificates (or a single SAN cert) is issued on the same day, then certificate expiry will prevent access to voicemail.

# Unified Messaging High Availability

Designing a Highly Available UM environment is an automatic side effect of designing a Highly Available Exchange 2013 environment – it's included in the box. The “every server is an island” concept, where everything needed to service a mailbox's needs is within the same server, apply here. Every mailbox has a local UM server available ready and waiting.

There are a number of design areas that do affect UM high availability though. Blindly designing a Database Availability Group with no concept of how this affects groups of users who access phone systems will make for a tricky implementation.

## DAG Design Implications

When designing your HA environment and determining user placement, you should take into account where users are planned to be placed.

One option, particularly suited to building-block infrastructure. If you are planning to build a number of Database Availability Groups and distribute users randomly, then you might wish to consider the UM Dial Plans you will create, and the associated Hunt Groups and Language Pack requirements. By distributing users randomly across a number of DAGs you may then need to associate every Exchange Mailbox server with every Hunt Group, and install every Language Pack on every single server. This will have implications for configuration in your phone system – and it will mean that you need to consider the bandwidth capacity from any IP gateway to any Exchange Server.

You may instead consider an isolated group approach and choose to align users in common Dial Plans with Database Availability Groups and size based on those numbers. This will allow you to keep the associated PBX configuration and network configuration tied to a smaller number of servers and keep maintenance simpler.

The downside to the isolated group approach verses the building block approach is that your design for each DAG may be different and reconfiguration and expansion is harder. This only really affects centralised deployments though, as most international deployments will typically be designed to use building-blocks for each country allowing user groups to be isolated and the benefits of building block infrastructure to be realised.

## Network Implications

After considering the type of approach you will use for design, the impact on the network infrastructure also must be taken into account.

In Exchange 2007 and 2010, a deployment could use a UM server located close to a local PBX. If the user's mailbox was in a different country it didn't matter as the PBX latency between the UM server and the PBX was all that mattered. The firewall rules between the PBX IP gateway and the user's mailbox server were also irrelevant too, because the UM server would talk to the Mailbox server rather than the PBX itself.

Exchange 2013 changes this as the PBX will first contact an Exchange 2013 server listed as one of its gateways. The UM Call Router service will then request the PBX contacts the relevant Exchange 2013 server hosting the UM Service for the user. The implication is that the bandwidth available and latency is crucial, as is the firewall rules between listed PBX servers, Client Access Servers and Mailbox Servers.

## Sizing Implications

It's not easy to predict how many concurrent calls to expect based on the type of data you will usually collect during a sizing exercise for Exchange servers – the number of messages received and sent per day

will not necessarily correlate to how many voicemails a user will receive, and how long the caller will spend recording a message.

To help with understanding how many concurrent calls to expect, look to your PBX administrator to help you understand the concurrency and per-user statistics for how often and how long voicemails are left.

In addition to existing statistics you will need to estimate overhead for other calls. If you are using auto-attendant features then you may expect a large increase in concurrent calls, especially if it replaces or supports main switchboard numbers for the organization.

Also consider the number of calls placed by users to their own mailboxes for Outlook Voice Access. Ensure a healthy overhead is added to cover these users.

Estimated data for call concurrency should be used in the Exchange Role Requirements calculator to ensure the split of users across servers is performed in line with making sure that each server does not exceed 100 concurrent calls, even in failure situations.

## Unified Messaging Summary

High availability for Unified Messaging depends not only on the Exchange servers but also the other components of the UM solution such as IP PBX and IP gateways.

Fortunately, a highly available Client Access and Mailbox server infrastructure achieves most of the requirements from the Exchange Server 2013 side of things, with load balancing and appropriate configurations for UM bringing it all together.

# Managing and Monitoring High Availability

When discussing high availability for an Exchange messaging environment, duplicating server roles, hardware or other components aren't enough.

It is also important to understand how to retain that high availability and resilience through the ongoing management and monitoring of the environment. Just like when you are securing an environment, a multi-layered approach yields the best results.

## Managed Availability

Let's start by explaining what Managed Availability is and what it does. The shortest way of describing it, would be to call it Exchange's built-in monitoring and remediation platform.

By continuously probing an Exchange server's health, Managed Availability will evaluate whether the server is still functioning optimally. If it notices there's something wrong, Managed Availability will kick off some remediating action to resolve the situation.

If for some reason the recovery action did not complete successfully or if the problem keeps returning, Managed Availability will 'call in' help from an Administrator by raising an alert in the Event Logs. The latter step is referred to as an escalation.



To borrow an analogy made by fellow MVP, Paul Robichaux, Managed Availability is like a doctor. If you go to the doctor's office, they will likely start by asking you questions on where you are having pain and do some more investigation to find out what is wrong (probing).

Then the doctor will review the results of his probes (monitor), and he will either write a prescription (remediation) or – in case they cannot solve the issue – forward you to a specialist for a second opinion (escalation).

Managed Availability runs on each Exchange 2013 server and consists of 2 services:

- Health Manager Service
- Health Manager Worker

## Health Manager Service

The Health Manager Service, MSExchangeHMHost.exe, is the service that is responsible for managing the worker process. It will keep track of the worker process and make sure that it does not become a single point of failure.

For instance, when the worker process should hang or isn't started, the Health Manager Service will automatically restart the service.

You can view the service either using PowerShell or in the Services console.

[PS] C:\Scripts>get-service MSExchangeHM*				
Status	Name	DisplayName		
Running	MSExchangeHM	Microsoft Exchange Health Manager		

When opening Microsoft's Process Explorer<sup>15</sup>, you can clearly see the relation between the Health Manager Service and the underlying child-process: the Health Manager worker process.

[-] MSExchangeHMHost.exe + MSExchangeHMWorker.exe	0.04	92 796 K	42 368 K	1564	Microsoft Corporation
	1.51	309 080 K	213 800 K	12456	Microsoft Corporation

<sup>15</sup> <http://technet.microsoft.com/en-us/sysinternals/bb896653.aspx>

## Health Manager Worker process

The Health Manager Worker process, MSExchangeHMWorker.exe, is the process that is responsible for executing the tasks related to the operations of Managed Availability. For instance, it will ensure that the probes are ran when they are supposed to run and it will also make sure that responders are fired if needed be.

MSExchangeFrontendTransport.exe	3812	Running	SYSTEM
MSExchangeHMHost.exe	1564	Running	SYSTEM
MSExchangeHMWorker.exe	12456	Running	SYSTEM
MSExchangeMailboxAssistants.exe	6192	Running	SYSTEM

One might think that the Health Manager Service is a single point of failure. After all: what would happen if the Health Manager Service isn't started or not working correctly? To mitigate this (potential) problem, Microsoft implemented a sort of "buddy"-system.

When the Health Manager Service starts up, it will send out a request to other servers in the organization to monitor its instance of the Health Manager Service. As such, the local machine is being 'monitored' by remote Exchange Servers which will restart the Health Manager Service when they notice if it's not behaving properly.

You can view which servers are being monitored by a specific Exchange Server by running the Get-ServerHealth cmdlet.

```
[PS] C:\> Get-ServerHealth -Server <servername> -HealthSet RemoteMonitoring | ?{$_ .Name -eq "HealthManagerObserverMonitor" } | ft Name,TargetResource
```

The TargetResource column shows which servers are being monitored by the Exchange server you specified in the –Server parameter.

## File Locations

Managed Availability stores its configuration information mainly in the <InstallDirectory>\Bin\Monitoring\Config folder:

D:\Program Files\Microsoft\Exchange\v15\Bin\Monitoring\Config				
	Name	Date modified	Type	Size
ds laces	RusPublisherWeb	6/05/2014 20:48	XML Document	7 KB
	Rws	6/05/2014 20:48	XML Document	1 KB
	ScopeMappings	6/05/2014 20:48	XML Document	60 KB
	SearchServiceContext	6/05/2014 20:48	XML Document	11 KB
	Security	6/05/2014 20:48	XML Document	1 KB
nts	SharedCacheServiceTest	6/05/2014 20:48	XML Document	2 KB
	SiteMailboxMaintenanceDef	6/05/2014 20:48	XML Document	2 KB
	SmtpProbes_Frontend	6/05/2014 20:48	XML Document	17 KB

The Health Manager Service will read information from these files when the service starts up as they contain settings for some of the probes, monitors and responders. Because the files are stored in XML format, they are easily readable for a human and one might be tempted to make some changes in order to ‘tweak’ Managed Availability.

Even though there’s nothing to prevent you from making a change, we strongly recommend against it. The reason being that a little mistake or typo in one of the files could cause the Health Manager Service to stop functioning.

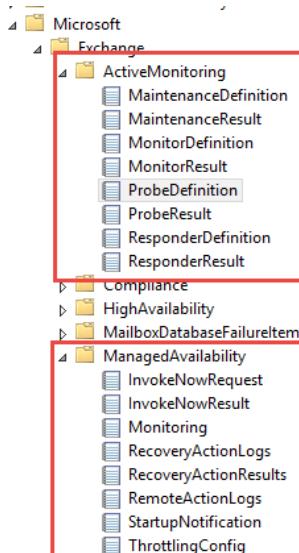
```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <Definition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..\..\..\ActiveMonitoring\WorkItemDefinition.xsd">
- <CustomWorkItem ExoDataCenter="false" FfoDataCenter="false" MonitoringMailbox="Cafe">
- <Probe Enabled="true" MaxRetryAttempts="1" TimeoutSeconds="60" RecurrenceIntervalSeconds="300" ComponentName="FrontendTransport" Name="OnPremisesInboundProxy" TypeName="Microsoft.Forefront.Monitoring.ActiveMonitoring.Smtp.Probes.SmtpConnectionProbe">
- <ExtensionAttributes>
- <WorkContext>
<SmtpServer>127.0.0.1</SmtpServer>
<Port>25</Port>
<HelloDomain>InboundProxyProbe</HelloDomain>
<MailFrom Username="inboundproxy@contoso.com"/>
<MailTo Select="All"/>
<Data AddAttributes="false">X-Exchange-Probe-Drop-Message:FrontEnd-CAT-250 Subject:Inbound proxy probe</Data>
<ExpectedConnectionLostPoint>None</ExpectedConnectionLostPoint>
</WorkContext>
</ExtensionAttributes>
```

Next to these XML files, Managed Availability also leverages Active Directory, the server’s local registry, as well as the Crimson Channel to store its data.

Active Directory and a server’s local registry are primarily used to store overrides. Overrides are used to temporarily disable specific probes or responders. When we get into more detail about these overrides, we’ll also talk more about where they are stored.

The Event Log’s Crimson Channel is where Managed Availability puts away most of its operational data. It will continuously update the Crimson Channel with what it’s currently doing so that an Admin can use that data to trace back what action Managed Availability recently took to mitigate a problem.

When storing data in the Event log, MA will do so in two places in the Crimson Channel:  
Microsoft/Exchange/ActiveMonitoring and Microsoft/Exchange/ManagedAvailability.



## HealthSets

Managed Availability has a set of a few hundred probes, monitors and responders which are designed to monitor the overall health of a server. In order to logically group all of these elements, probes, monitors and responders are logically grouped together in a HealthSet.

Each HealthSet contains a combination of probes, monitors and responders specific to a workload, feature or service within Exchange. Depending on whether you have deployed a single-role server or a multi-role server, you might have different Healthsets active on a server. HealthSets specific to a function the mailbox role is responsible for won't be present on a CAS-only server and vice versa.

To view what HealthSets are present on a given server, you can run the Get-ServerHealth cmdlet.

Name	Count
ActiveSync	2
ActiveSync.Protocol	7
ActiveSync.Proxy	1
AD	18
AMADError	2
AMEUS	4
AntiSpam	6
.....	

**Note:** For more information about the HealthSets that are available refer to TechNet<sup>16</sup>.

## Probes

Probes are used to gather information on the health of a service, workload or component in Exchange. There are different types of probes, each having its own specific purpose. There are two type of probes; active probes, and passive probes.

Active probes will pro-actively test a component or set of components by executing some sort of component self-test or even a synthetic transaction. The synthetic transactions are called Customer Touch Point (CTP) probes and are most commonly used to test components within the Client Protocols HealthSet like OWA, ECP or Outlook Anywhere. The synthetic transaction that is being executed as part of such a probe mimics a specific user action such as logging into OWA.

---

<sup>16</sup> [http://technet.microsoft.com/en-us/library/dn195906\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/dn195906(v=exchg.150).aspx)

Passive probes monitor the value of specific counters such as performance counters or disk space counters. These probes don't actively go out to test the feature, but rely on the information that is returned by the counter itself.

The combination of active and passive probes creates a detailed view of a server's health. After all, a component that isn't working appropriately can be caused by e.g. a lack of disk space and not necessarily because the component itself is broken.

Probes don't evaluate results. That task is performed by Monitors, which are discussed later in this chapter.

There are different ways in which you can check which probes are currently active on your Exchange server. Given that Managed Availability heavily uses the Crimson Channel in the Event Logs, this is a good place to start looking.

The probes that are defined on your Exchange server will be stored in the **Microsoft/Exchange/Active Monitoring/ProbeDefinition** channel. Each time the Health Manager Service starts up, the probes will be written there.

The information that is displayed by default won't be of much help though. The useful parts are stored as XML data inside the details of the event:

If you know what you are looking for, the Event Viewer can be a good place to start with. However, it is not easy to find something in there if you don't know what you are looking for. PowerShell offers a better alternative at disclosing specific information in a structure way. As you will learn, the syntax to get that information isn't as easy as it should be, but the more you use it, the more sense it will make.

First, we need to grab the events from the Crimson Channel. Because the data is stored in XML, we also need to make sure that we tell PowerShell to treat the data as XML.

```
[PS] C:\>$Probes = (Get-WinEvent -Computer localhost -LogName "Microsoft-Exchange-ActiveMonitoring/ProbeDefinition" | %{{[xml]$_.toXML()}}).event userdata.eventXML
```

Now that the probes are stored in memory (using the \$probes variable), we can query the variable and try to make some sense of the data. For instance, if you want to list all the probes on the system and which resource is being probed, run the following command.

```
[PS] C:\>$Probes | ft Name,TargetResource
```

Name	TargetResource
ComplianceOutlookLogonToArchiveRpcCtpProbe	DB04
ComplianceOutlookLogonToArchiveRpcCtpProbe	DB03
ComplianceOutlookLogonToArchiveRpcCtpProbe	DB02
ComplianceOutlookLogonToArchiveRpcCtpProbe	[null]
OutlookRpcCtpProbe	DB04
OutlookRpcCtpProbe	DB03

As you will notice, probes can exists multiple times - once for each target. Database-specific probes will therefore show up for each database that exists on a server.

Some probes, like the CTP probes, require a user account (mailbox) to run. To see which account a specific test is using, you can add the Account parameter to the previous command. If we were to narrow down the results for the OWA CTP probe, you would run the following command.

```
[PS] C:\> [PS] C:\>$Probes | ?{$_ .Name -eq "OwaDeepTestProbe"} | ft Name,TargetResource,Account
```

Name	TargetResource	Account
OwaDeepTestProbe	DB04	HealthMailbox04aa7805d70f436783751e8e3edc7938@exchange2013demo.com
OwaDeepTestProbe	DB03	HealthMailbox3d22ed6d203c4bea851c14dfbcf02055@exchange2013demo.com
OwaDeepTestProbe	DB01	HealthMailboxbc5134a4e90c45cd895cbb79fe0a3792@exchange2013demo.com
OwaDeepTestProbe	DB02	HealthMailbox51a0ac309ddf491387a325a53a8dc0b0@exchange2013demo.com

The Health Manager Service automatically creates these health mailboxes and will do so for each mailbox database in the environment.

Using the following command, you can see which health mailboxes have been created in your environment.

```
[PS] C:\>Get-Mailbox Health* -Monitoring
```

Name	Alias	ServerName	ProhibitSendQuota
HealthMailbox04aa7805d...	HealthMailbox04aa...	sydex1	Unlimited
HealthMailbox13c7c528b...	HealthMailbox13c7...	sydex1	Unlimited
HealthMailbox14f0b7400...	HealthMailbox14f0...	sydex1	Unlimited
HealthMailbox190210799...	HealthMailbox1902...	sydex1	Unlimited

## Monitors

Managed Availability's monitors are responsible for monitoring and evaluating the results which are returned by the probes. These results can either be a success or failure of a certain probe, or the value of a counter which is returned by a probe. For counter values, a monitor will evaluate the returned value against a pre-defined threshold.

Depending on what kind of result is returned a monitor can either initiate a responder which will try to fix the issue or it can escalate the issue for an admin to take a look at. When escalating an issue to an admin, Managed Availability will post an entry in the Event Log. More specifically, it will do so in the **Microsoft\Exchange\ActiveMonitoring\ResponderResult** channel.

As you might have noticed by the location where an escalation is written into the Event Log, an escalation is actually a responder's work too. Compared to the other responders, this one just posts a message in the event log.

Unlike in earlier versions, the Exchange Management Pack for System Center Operations Manager now doesn't perform any of the probing. Instead it relies on Managed Availability to do 'the right thing'. The Management Pack will pick up any escalations that Managed Availability writes into the Event Logs and use these to throw a warning in the dashboard. When you take a look at the definition of a specific escalation responder, you will notice it contains the escalation message which SCOM will also show you. For instance, the definition for the MailboxTransportDeliveryServiceStartFailureEscalateResponder (which is fired when the Transport Delivery Service cannot be started repeatedly) contains the following text:

```
<EscalationSubject>Mailbox Transport delivery service did not successfully initialize.</EscalationSubject>
<EscalationMessage>{Probe.Error} Probe Exception: '{Probe.Exception}' Failure Context:
'{Probe.FailureContext}' Execution Context: '{Probe.ExecutionContext}' Probe Result Name:
'{Probe.ResultName}' Probe Result Type: '{Probe.ResultType}' Monitor Total Value: '{Monitor.TotalValue}'
Monitor Total Sample Count: '{Monitor.TotalSampleCount}' Monitor Total Failed Count:
'{Monitor.TotalFailedCount}' Monitor Poisoned Count: '{Monitor.PoisonedCount}' Monitor First Alert
Observed Time: '{Monitor.FirstAlertObservedTime}'</EscalationMessage>
<EscalationTeam>E15Transport</EscalationTeam>
```

This information is easily retrievable by looking at a specific responder's definition in the Microsoft\Exchange\ActiveMonitoring\ResponderDefinition Crimson Channel.

**Real World:** Because Managed Availability writes its escalations to the Event Log, the usefulness of this information will depend on your monitoring solution. If your monitoring system doesn't have the capability to alert you to Managed Availability escalations, then you may need to invest some time in a custom scripting solution instead.

Monitors have multiple states, some of which are directly exposed to the admin. Typically, you will notice that a monitor is either "Healthy" or "Unhealthy".

Both terms are self-explanatory, but there are also some additional states which you might sometimes encounter, albeit less frequently. For instance, during the first 60 seconds that a monitor is unhealthy it will not be reported as being so, instead the monitor will be in the "Degraded" state. After the monitor stays unhealthy for more than 60 seconds it will turn into "Unhealthy".

Other states that you might encounter are:

- Disabled - When a monitor is disabled on purpose
- Repairing - When an admin purposely set a monitor into the Repairing state when working on a previously reported issue. This ensures that either Managed Availability or other admins can correlate other issues reported by the same monitor at the same time an admin is working on resolving the issue.
- Unavailable - This happens when the Health Manager Service cannot determine the state of a monitor; for instance, because it's not responding anymore.

To switch the state from a specific monitor, such as the AutodiscoverProxy monitor, into the "Repairing" state, an Admin has to manually run a PowerShell command first.

```
[PS] C:\>Set-ServerMonitor -Server SYDEX1 -Name AutodiscoverProxy -TargetResource MSEExchangeAutodiscoverProxy -Repairing $true
```

Once that is done, the Get-ServerHealth will report an AlertValue of "Repairing" instead of "Unhealthy". After the admin has finished working on the issue they can switch the monitor back into its normal state by issuing the above command again.

Existing monitors can either be viewed in the Event Log in the **Microsoft\Exchange\ActiveMonitoring\MonitoringDefinition** Crimson Channel or through PowerShell.

```
[PS] C:\>$Monitors = (Get-WinEvent -Computer localhost -LogName "Microsoft-Exchange-ActiveMonitoring/MonitorDefinition" | %{{[xml]$_.toXML()}}).event userdata.eventXML
```

Using the \$Monitors variable, you can now find out all sorts of information from the monitors on the system. By executing the following command, you can find out what actions a monitor will be kicked off as a monitor changes its state.

```
[PS] C:\>$Monitors | ?{$_._StateTransitionsXML -ne "[null]"} | fl Name,StateTransitionsXML

Name          : MSClassificationEngineErrorsMonitor
StateTransitionsXml : <StateTransitions>
                    <Transition ToState="Unhealthy" TimeoutInSeconds="0" />
                </StateTransitions>

Name          : MicrosoftEngineErrorsMonitor
StateTransitionsXml : <StateTransitions>
                    <Transition ToState="Degraded" TimeoutInSeconds="0" />
                </StateTransitions>
```

For each of the stages that are defined in the StateTransitionsXML parameter for a given monitor, there will be a matching responder. As a monitor changes into the “Degraded” state, a responder will be executed.

After 420 seconds, and if the monitor changed into the “Unhealthy” state, the responder that is configured to run when this particular monitor is in the “Unhealthy” state will then be invoked. When the monitor then continues to stay in the “Unhealthy” state another responder will be executed, only after 900 seconds. And this will continue until the monitor reaches the “Unrecoverable” state after which the escalation responder will post an entry into the Event Log.

This mechanism ensures that a specific recovery action is not being executed over and over without result. However, if a responder’s actions turn the monitor back into the “Healthy” state, the process starts over again.

When a component repeatedly fails, but a responder manages to bring the monitor back into the “Healthy” state each time, the throttling of responders will ensure that a specific action does not jeopardize the operation of that server. We will discuss the throttling of responders later in this chapter.

**Real World:** Although monitors have various states, most of these states are used for internal purposes only. That is why when we are referring to a monitor or a server’s component state we typically report it to be either *Healthy* or *Unhealthy*.

## Monitoring Overrides

Monitoring Overrides can be used to temporarily change a monitor’s behavior or to temporarily disable it. The reason why it is only temporary is because you always have to specify the **-Duration** parameter

with the cmdlet. If you need the override to still be in place after the override duration expires, you will need to manually re-apply it.

There are two types of overrides; global and local.

Global Overrides are used to change the behavior of a monitor on all servers at the same time. When you configure a Global Override, it is stored in Active Directory and therefore it might take a while before all servers in your environment will pick up the newly configured override. The Global Overrides are stored in the Exchange settings of the Configuration Partition.

To configure a Global Override, you should use the Add-GlobalMonitoringOverride command. The following example, changes the threshold for the disk space monitor on Database Volumes.

```
[PS] C:\> Add-GlobalMonitoringOverride -Identity MailboxSpace\DatabaseSizeMonitor -ItemType Monitor -PropertyName ExtensionAttributes.DatabaseLogsThreshold -PropertyValue 10GB -Duration 60.00:00:00
```

After running the command, run ADSIEDit and open the Configuration Partition. Next, navigate to the the “CN=Monitor,CN=Overrides,CN=Monitoring Settings,CN=OrgName,CN=Microsoft Exchange,CN=Services,CN=Configuration” container.

ADSIEDit will show you the global override that you have configured.

**Warning:** Always be careful when using ADSIEDit! Making changes in ADSIedit is not supported by Microsoft. You should always consider ADSIEDit as a “look, but don’t touch” tool.

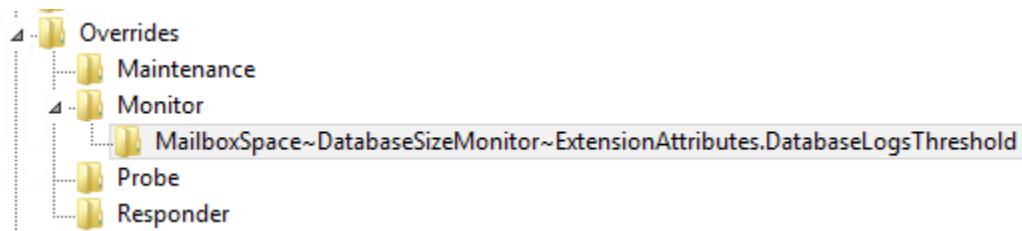
Local Overrides only apply to the server for which you configure the override. A local override is stored in a server’s local registry in the following location:

*HKEY\_LOCAL\_MACHINE\Software\Microsoft\ExchangeServer\v15\ActiveMonitoring\Overrides*

Configuring Local Overrides is very similar to creating Global Overrides, except you now use the Add-ServerMonitoringOverride command instead.

```
[PS] C:\> Add-ServerMonitoringOverride -Identity MailboxSpace\DatabaseSizeMonitor -ItemType Monitor -PropertyName ExtensionAttributes.DatabaseLogsThreshold -PropertyValue 10GB -Duration 60.00:00:00
```

After running the command, this is how the server's registry will show the override.



Using the Get-GlobalMonitoringOverride or Get-ServerMonitoringOverride cmdlets, you can query either Active Directory or a specific server for any configured overrides. That way, you don't necessarily have to fiddle around with ADSIedit or the Registry Editor to retrieve the information.

```
[PS] C:\>Get-ServerMonitoringOverride SYDEX1
```

## Responders

Responders are initiated by a monitor in response to a probe result. Each responder has a specific task which can vary from restarting a service to bugchecking (rebooting) a server. Some of the more common responder types include:

- Service Restart Responder - (Re)starts a service
- Failover Responder - Initiates a database or server failover
- Online/Offline Responder - Responsible for switching the state of a component online/offline
- Reset AppPool Responder - (Re)starts an application pool in IIS
- Bugcheck Responder - Executes a bugcheck which causes a reboot of the server
- Escalate Responder - Writes an event into the event log to escalate the issue to an admin

When the Health Manager service starts up, it will write into the Event Log which Responders exist on the system. Just like the probes, you can use PowerShell to fetch those responders.

```
[PS] C:\>$Responders = (Get-WinEvent -Computer localhost -LogName "Microsoft-Exchange-ActiveMonitoring/ResponderDefinition" | %{{[xml]$_.toXML()}}).event userdata.eventXML
```

Using the \$Responders variable, you can now query it to display a specific responder or get specific information.

For instance, the following command will display all responders on the system and display its name and throttling information.

```
[PS] C:\>$Responders | ft Name,ThrottlePolicyXML
```

## Throttling

Just like the StateTransitionsXML on a monitor, which defines when a specific responder should be executed to avoid that certain responders are executed over-and-over again, responders themselves are also throttled. This is to ensure that when a responder's action does bring a monitor back into a *Healthy* state, and then another failure occurs, that the responder does not repeatedly execute without ever alerting the administrators so that the underlying issue can be resolved.

The command we used in the previous example will display the Throttling information per responder. The throttling configuration will look similar to this.

```
<ThrottleEntries>
  <RecycleApplicationPool ResourceName="MSExchangeAutoDiscoverAppPool">
    <ThrottleConfig Enabled="True" LocalMinimumMinutesBetweenAttempts="60"
      LocalMaximumAllowedAttemptsInOneHour="-1" LocalMaximumAllowedAttemptsInADay="1"
      GroupMinimumMinutesBetweenAttempts="60" GroupMaximumAllowedAttemptsInADay="4" />
  </RecycleApplicationPool>
</ThrottleEntries>
```

This code belongs to the responder which will recycle the AutoDiscover App Pool whenever executed. The LocalMinimumMinutesBetweenAttempts value, tells us that the responder can only restart the App Pool once an hour.

Each of the ThrottleEntries for a responder contain Local throttling settings and Group throttling settings. The difference between both is that the Group settings only apply if the server is part of a Database Availability Group. In the previous code example the GroupMaximumAllowedAttemptsInADay value ensures that the AutoDiscover App Pool would not be restarted more than four times per day when the server is part of a DAG.

While restarting an Application Pool might seem like a trivial action, one for which throttling might seem less important, the effects of throttling are much more visible when it comes to bugchecking a server. Typically the bugcheck responders are limited to restarting a server only once a day, which prevents the undesirable scenario of a server being rebooted multiple times per day.

## Server Component States

When Managed Availability decides that a component is unhealthy because one or more probes failed it will trigger an "Offline Responder". This responder will mark the failed component as "Offline" or "Inactive"; very similarly to how a monitor is set to "Unhealthy".

When a component is marked as being “Inactive”, that component is not able to handle any workload. Because the Server Component States are also exposed to other servers in the environment, the other servers will also disregard that Exchange Server’s failed component for as long as it is inactive. For example, a Mailbox Server in a Database Availability Group for which the HubTransport Component is inactive, will not be used for routing messages within the DAG.

When you come to think of it this approach is pretty smart, and increases the resiliency of your entire solution. Instead of having other servers rely on a component which might (or might not) be broken, now the other Exchange servers won’t attempt to use the broken component, which therefore avoids a potential failure in the system. In turn, this results in a higher availability rate for the *service* (e.g. mail delivery) – despite the failure of a single component directly related to that service.

To retrieve the component states of a specific server, run the following PowerShell command.

Server	Component	State
SYDEX1.exchange2013demo.com	ServerWideOffline	Active
SYDEX1.exchange2013demo.com	HubTransport	Active
SYDEX1.exchange2013demo.com	FrontendTransport	Active
SYDEX1.exchange2013demo.com	Monitoring	Active
SYDEX1.exchange2013demo.com	RecoveryActionsEnabled	Active
SYDEX1.exchange2013demo.com	AutoDiscoverProxy	Active
SYDEX1.exchange2013demo.com	ActiveSyncProxy	Active
SYDEX1.exchange2013demo.com	EcpProxy	Active
SYDEX1.exchange2013demo.com	EwsProxy	Active
SYDEX1.exchange2013demo.com	ImapProxy	Active
SYDEX1.exchange2013demo.com	OabProxy	Active
SYDEX1.exchange2013demo.com	OwaProxy	Active
SYDEX1.exchange2013demo.com	PopProxy	Active
SYDEX1.exchange2013demo.com	PushNotificationsProxy	Active
SYDEX1.exchange2013demo.com	RpsProxy	Active
SYDEX1.exchange2013demo.com	RwsProxy	Active
SYDEX1.exchange2013demo.com	RpcProxy	Active
SYDEX1.exchange2013demo.com	UMCallRouter	Active
SYDEX1.exchange2013demo.com	XropProxy	Active
SYDEX1.exchange2013demo.com	HttpProxyAvailabilityGroup	Active
SYDEX1.exchange2013demo.com	ForwardSyncDaemon	Active
SYDEX1.exchange2013demo.com	ProvisioningRps	Active
SYDEX1.exchange2013demo.com	MapiProxy	Active
SYDEX1.exchange2013demo.com	EdgeTransport	Active
SYDEX1.exchange2013demo.com	HighAvailability	Active
SYDEX1.exchange2013demo.com	SharedCache	Active

**Note:** The list of components does not match the Exchange Services 1-on-1. Instead components represent certain workloads in Exchange and might consists of one or more underlying services.

A component typically can be in one of two states: “Active” or “Inactive”. Only the HubTransport and FrontEndTransport components can be in a third state: “Draining”.

Draining means that the service is in process of being made inactive, but that it is allowed to finish processing messages that are currently in queue. This is particularly useful when performing maintenance on a server, which is discussed later in this chapter.

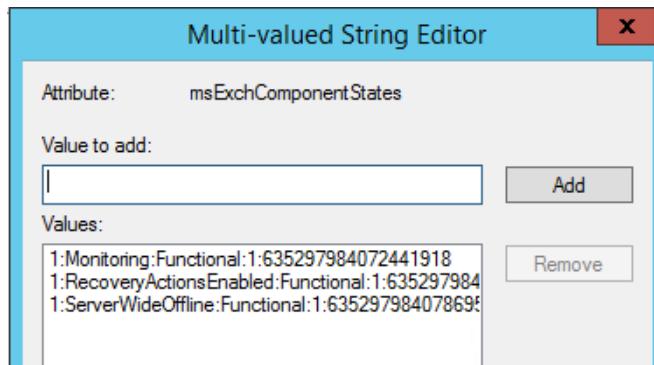
Unlike monitors which can only have their state changed by the Health Manager Service, there are multiple reasons why a component can be marked as “Inactive”. These are identified as Requesters.

Requesters are used to mark who or what requested the state change of a specific component. In total, there are five different requesters:

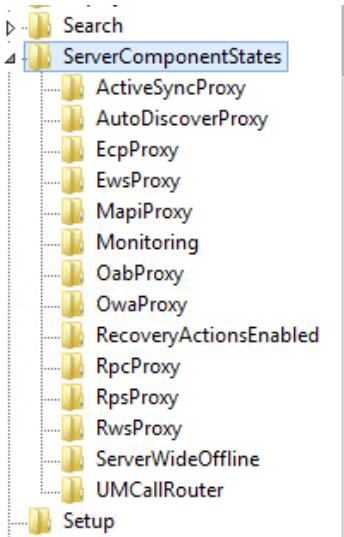
- HealthAPI (Managed Availability)
- Maintenance
- Sidelined
- Functional
- Deployment

The Get-ServerComponentState command unfortunately does not immediately expose the requester that initiated the state change. Before we dive deeper into getting that information, it is important to know that the information regarding a server’s component states is stored in two locations; Active Directory and the server’s registry.

The Active Directory information allows other Exchange Servers to query a server’s health without having to query the server itself. The component states are saved as a property of the server object in the Exchange Configuration container. The object is called **msExchComponentStates**.



On the local server, the settings are stored in the registry in  
**HKLM\Software\Microsoft\ExchangeServer\v15\ServerComponentStates**.



To retrieve the state and the requester of that state for a specific component using the Get-ServerComponentState cmdlet.

For the Local State(s):

```
[PS] C:\>(Get-ServerComponentState -Identity SYDEX1 -Component RpcProxy).LocalStates
```

Requester	State	Timestamp	Component
HealthApi	Active	7/11/2014 8:50:20 AM	RpcProxy

For the Remote (Active Directory) State(s):

```
[PS] C:\>(Get-ServerComponentState -Identity SYDEX1 -Component OwaProxy).RemoteStates
```

Requester	State	Timestamp	Component
Maintenance	Active	21/10/2014 12:07:50 PM	OwaProxy

#### *Changing a component's state*

As an administrator you might want to perform work on a component such as troubleshooting, maintenance, or installing new software that interacts with that component.

Although, in theory, you could use any of the earlier mentioned requesters to change a component's state, it's advised to use the *Maintenance* requester to do so.

For example, to take the FrontEndTransport component on a server called SYDEX1 offline, you would use the following command.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component FrontEndTransport -State Inactive -Requester Maintenance
```

Similarly, to return the component back into its original (*Active*) state, run the following command.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component FrontEndTransport -State Active -Requester Maintenance
```

It is possible for multiple requesters to have requested a certain component to be “Inactive”. For example, a component may have already been taken offline by Managed Availability, and an administrator has also requested to take that same component offline while performing maintenance.

In such case, you will notice that you might end up with multiple entries when requesting the details of a component’s state.

```
[PS] C:\>(Get-ServerComponentState -Identity SYDEX1 -Component FrontEndTransport).LocalStates
```

Requester	State	Timestamp	Component
Maintenance	Inactive	7/11/2014 9:01:56 AM	FrontEndTransport
HealthAPI	Inactive	7/11/2014 8:15:23 AM	FrontEndTransport

When the admin has finished working on the server, they would request the component to be brought back online using the Set-ServerComponentState cmdlet. However, when querying the component afterwards, the component would still be offline because there is still one requester left which requested it to be taken offline.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component FrontEndTransport -State Active -Requester Maintenance
```

```
[PS] C:\>(Get-ServerComponentState -Identity SYDEX1 -Component FrontEndTransport).LocalStates
```

Requester	State	Timestamp	Component
HealthAPI	Inactive	7/11/2014 8:15:23 AM	FrontEndTransport

As a result, a component will only be brought back online (or “Active”) when no more entries exists of a request to take it offline. It is possible for an admin to force a component into the Active state by manually removing any remaining requests.

In this particular case, Managed Availability (HealthAPI) was the one who requested the component to be taken offline. The admin can issue the following command and remove Managed Availability's request for the component to be taken offline, therefore forcing it to become "Active" again.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus * | where {$_.ContentIndexState -eq "Failed"} | Update-MailboxDatabaseCopy -CatalogOnly
```

**Warning:** even though the admin can remove the override, it should be noted that Managed Availability would not have taken that action without a good reason in the first place. Before just switching a component back online, make sure that you fully investigate why it was inactive in the first place.

One of the server components is named "RecoveryActionsEnabled". This component can be set to an "Inactive" state to stop Managed Availability from taking any action at all on the server. This would be useful in test lab environments and when you are still configuring a newly installed server, however it is not recommended for use on production servers.

For more information on this setting refer to [Exchange MVP Brian Reid's blog post here](#)<sup>17</sup>.

## Performing Server Maintenance

Routinely patching/updating servers is part of an Exchange administrator's job. Typically this is a relatively straightforward action to perform. When you are dealing with highly available systems, some of which might be tied to very tight Service Level Agreements, the task suddenly becomes less trivial.

Because Exchange 2013, more than before, interacts with its siblings within the organization, it's important that you let other servers know when one is temporarily unavailable because you are performing work on it.

One way would be to let the Health Manager Service take control of it as components would eventually be brought offline. The thing is that there might be a delay between the moment you start working on a server and when the components are finally brought offline. In this time, other Exchange servers might still send requests to that server. Some requests might then fail possibly affecting the service availability. And that's exactly what we're trying to avoid by having a highly available system.

---

<sup>17</sup> <http://www.c7solutions.com/2012/10/placing-exchange-2013-into-maintenance-html>

## Putting a Server into Maintenance Mode

The answer to the problem discussed here above lies in the so-called Maintenance Mode. In fact, Exchange doesn't have a Maintenance Mode-switch. Instead it is referred to as the result of a series of steps the admin goes through to temporarily ensure a server is properly taken out of service in the organization. Depending on whether your server is part of a Database Availability Group or not, your actions may vary.

Typically, the actions you have to take are:

1. Drain and stop the Transport services
2. Drain and stop Unified Messaging calls
3. Suspend the server in the cluster so that replication is halted for this server (if the server is a member of a DAG)
4. Make sure that no new client/server requests are processed
5. Verify that a server has successfully been placed into maintenance mode

### *Drain and Stop the Transport Services*

Before taking Transport services entirely offline, it is important to process messages that might still exist in the server's queue. This is true only for the HubTransport component which exists on the Mailbox Server role. The FrontEndTransport component on the Client Access Server role does not queue messages, therefore it has no queue to drain.

For Mailbox servers start by putting the HubTransport component into a "Draining" state. This ensures that the HubTransport component will continue to process messages that are already in the queue, but it will not accept new ones.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component HubTransport -State Draining -Requester Maintenance
```

To speed up the draining process, it is a good idea to redirect remaining messages in the queue to another server so that this other server can process them instead. After all, messages can stay in a queue for several days as the server keeps retrying to send them. This would seriously delay your maintenance if you did not redirect them to another server.

```
[PS] C:\>Redirect-Message -Server SYDEX1 -Target SYDEX2.exchange2013demo.com
```

To verify whether the queues are empty on a server, run the following command.

```
[PS] C:\>Get-Queue -Server SYDEX1 | ?{$_._Identity -notlike "*\Poison" -and $_._Identity -notlike "*\Shadow\*"} | Select MessageCount
```

If the returned value is 0, you can safely proceed to the next step. If not, wait until the queues are empty.

**Note:** In this example, the Shadow Redundancy and Poison queues were ignored. This because the Poison queue does not get processed and would never reach the 0 count. The Shadow Redundancy queue is also less important in this case and therefore it can also be ignored safely.

For Client Access servers the following command will place the FrontEndTransport component into the “Inactive” state.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component FrontEndTransport -State Inactive -Requester Maintenance
```

#### *Drain and Stop Unified Messaging Calls*

Similar to the HubTransport component, we should place the UMCallRouter component into the “Draining” state so that it stops accepting new calls.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component UMCallRouter -State Draining -Requester Maintenance
```

Unlike the HubTransport component the active calls can't be redirected, and you will need to wait for them to complete before proceeding with your maintenance. You can use Get-UMActiveCalls to monitor the servers for active calls.

```
[PS] C:\>Get-UMActiveCalls -Server SYDEX1
```

#### *Suspending the Cluster Node*

For Mailbox servers that are members of a DAG suspending the cluster node ensures that all active databases are moved to another server in the DAG and that this server is temporarily not used as a target to move databases to either.

Additionally this process also makes sure that any critical roles that might exists on this node (such as the Primary Active Manager) are moved to another DAG member.

First, start by suspending the node in the cluster.

```
[PS] C:\>Suspend-ClusterNode SYDEX1  
Name ID State  
--- -- -----  
SYDEX1 2 Paused
```

Next, move all active databases to other DAG members. This process will automatically take care of activating the databases on other servers. The decision where to mount the databases is made by the Active Manager and the Best Copy and Server Selection process that was discussed in the Mailbox chapter of this guide.

```
[PS] C:\>Set-MailboxServer SYDEX1 -DatabaseCopyActivationDisabledAndMoveNow $true
```

Then, run the following command to block database activation on the server.

```
[PS] C:\>Set-MailboxServer SYDEX1 -DatabaseCopyAutoActivationPolicy Blocked
```

#### *Making Sure That No New Client/Server Requests are Processed*

The last step is to disable all remaining components. Instead of doing so component-per-component, there is a special component called ServerWideOffline which will take care of all that for you.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component ServerWideOffline -State Inactive -Requester Maintenance
```

## Verifying That a Server Has Successfully Been Placed into Maintenance Mode

Once you have run through the previous steps, you should verify whether the server is now successfully placed into Maintenance Mode. You can do this by running the following commands.

First, verify that all components are “Inactive”, except for Monitoring and RecoveryActionsEnabled.

Server	Component	State
SYDEX1.exchange2013demo.com	ServerWideOffline	Inactive
SYDEX1.exchange2013demo.com	HubTransport	Inactive
SYDEX1.exchange2013demo.com	FrontendTransport	Inactive
SYDEX1.exchange2013demo.com	Monitoring	Active
SYDEX1.exchange2013demo.com	RecoveryActionsEnabled	Active
SYDEX1.exchange2013demo.com	AutoDiscoverProxy	Inactive
SYDEX1.exchange2013demo.com	ActiveSyncProxy	Inactive
SYDEX1.exchange2013demo.com	EcpProxy	Inactive
SYDEX1.exchange2013demo.com	EwsProxy	Inactive
SYDEX1.exchange2013demo.com	ImapProxy	Inactive
SYDEX1.exchange2013demo.com	OabProxy	Inactive
SYDEX1.exchange2013demo.com	OwaProxy	Inactive
SYDEX1.exchange2013demo.com	PopProxy	Inactive
SYDEX1.exchange2013demo.com	PushNotificationsProxy	Inactive
SYDEX1.exchange2013demo.com	RpsProxy	Inactive
SYDEX1.exchange2013demo.com	RwsProxy	Inactive
SYDEX1.exchange2013demo.com	RpcProxy	Inactive
SYDEX1.exchange2013demo.com	UMCallRouter	Inactive
SYDEX1.exchange2013demo.com	XropProxy	Inactive
SYDEX1.exchange2013demo.com	HttpProxyAvailabilityGroup	Inactive
SYDEX1.exchange2013demo.com	ForwardSyncDaemon	Inactive
SYDEX1.exchange2013demo.com	ProvisioningRps	Inactive
SYDEX1.exchange2013demo.com	MapiProxy	Inactive
SYDEX1.exchange2013demo.com	EdgeTransport	Inactive
SYDEX1.exchange2013demo.com	HighAvailability	Inactive
SYDEX1.exchange2013demo.com	SharedCache	Inactive

Secondly, make sure that no databases are currently active on the server. You should see no database copies with a status of “Active” in the output of the following command.

Name	Status	CopyQueueLength	ReplayQueueLength	LastInspectedLogTime	ContentIndexState
DB01\SYDEX1	Healthy	0	0	7/11/2014 9:56:43 PM	Healthy
DB03\SYDEX1	Healthy	0	0	7/11/2014 9:55:38 PM	Healthy
DB04\SYDEX1	Healthy	0	0	7/11/2014 9:55:04 PM	Healthy
DB02\SYDEX1	Healthy	0	850	7/11/2014 9:59:38 PM	Healthy

Next, verify that the cluster node is paused in the cluster.

[PS] C:\>Get-ClusterNode SYDEX1		
Name	ID	State
SYDEX1	2	Paused

Last, but not least, make sure that the transport queues are empty. Although you have already done this previously, it doesn't hurt to double-check. This way you also verify that the Transport Service isn't accepting new messages.

```
[PS] C:\>Get-Queue -Server SYDEX1 | ?{$_._Identity -notlike "*\Poison" -and $_._Identity -notlike "*\Shadow\*"} | Select MessageCount
```

## Taking a Server Out of Maintenance Mode

When you are ready to take the server out of Maintenance Mode and put it back into its original state, you basically need to invert the changes that led the server to be in the Maintenance Mode. These steps include:

1. Re-enabling server components
2. Re-enabling the UM Call Router
3. Resuming the cluster node (if member of a DAG)
4. Re-enabling the Transport service(s)

### *Re-Enabling Server Components*

To re-enable the different components on the server, run the following command.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component ServerWideOffline -State Active -Requester Maintenance
```

### *Re-Enabling the Unified Messaging Call Router*

The following command will re-enable the UM Call Router component.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component UMCallRouter -State Active -Requester Maintenance
```

### *Resuming the Cluster Node*

For Mailbox servers that are DAG members the server needs to be “un-paused” in the cluster. You can do this with the following command.

```
[PS] C:\>Resume-ClusterNode SYDEX1
```

Name	ID	State
---	--	-----
SYDEX1	2	Up

Next, the server should be configured to allow active databases to be moved and mounted.

```
[PS] C:\>Set-MailboxServer SYDEX1 -DatabaseCopyActivationDisabledAndMoveNow $false  
[PS] C:\>Set-MailboxServer SYDEX1 -DatabaseCopyAutoActivationPolicy Unrestricted
```

### *Resuming the Transport Services*

The last step is to re-enable the Transport services. For Mailbox servers run the following commands.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component HubTransport -State Active -Requester Maintenance  
[PS] C:\>Invoke-Command -ComputerName SYDEX1 {Restart-Service MSExchangeTransport}
```

For Client Access servers run the following commands.

```
[PS] C:\>Set-ServerComponentState SYDEX1 -Component FrontEndTransport -State Active -Requester Maintenance  
[PS] C:\>Invoke-Command -ComputerName SYDEX1 {Restart-Service MSExchangeFrontEndTransport}
```

If the server runs both Mailbox and Client Access server roles then of course you would run both sets of commands.

**Note:** The process of putting an Exchange server into maintenance mode is tedious and lengthy which leaves room for error. On top of that, it is likely that you will have to execute these steps regularly.

Although I have no doubt that you will become quite experienced at executing these steps, automating your way through this process can be quite helpful. Exchange Server MVP Michael Van Horenbeeck has written a PowerShell script which takes care of the different steps outlined earlier.

- [Script: Putting Exchange Server 2013 into Maintenance Mode<sup>18</sup>](#)

The script will not only execute these steps, but it will also carry out some additional checks to make sure that every step can be executed successfully.

---

<sup>18</sup> <http://vanhybrid.com/2013/11/28/script-putting-exchange-server-2013-into-maintenance-mode/>

# Server Recovery

Despite the many features that are built into Exchange 2013 to prevent a server failure, there's little you can do to prevent hardware failures. This is just one of the few scenarios which might force you to recover a failed server. Before diving into the recovery process itself, let's have a look at the implications of where Exchange stores its data on that process.

## Storage locations

An Exchange server will store its data in various locations. To keep things simple, let's divide the two main types of data into the following categories:

- Configuration Data
- User Data (e.g. mailbox data)

### *Configuration Data*

Ever since Exchange Server 2007, Exchange stores the majority of its configuration settings in Active Directory. While it greatly increases the dependency on Active Directory, it does remove the need to backup settings per server as they are now part of the backup and availability process in Active Directory. The biggest benefit of having the configuration stored in a single, central, location is that it can easily be queried – also during a restore process.

Not all configuration data is fit to be stored in Active Directory. For example, the certificates which are used by Exchange are stored in the server's local registry and need to be re-imported once the recovery procedure has been completed. Along with the certificates, there are some other things – mostly customizations - which aren't stored in Active Directory either. The following list depicts some of these items, but is definitely not exhaustive:

- Customizations like 3<sup>rd</sup>-party transport agents or ISAPI filters
- Custom UM audio prompts
- Custom Outlook Web App files (themes etc.)
- Customizations to any of the server-side files

### *User Data*

User data itself is stored within mailbox databases that reside on the Exchange server. While the configuration data about the mailbox database is stored in Active Directory (for example, the file path where the database is stored), the databases files and the contents they contain are stored in the file system on the Mailbox servers.

## Backup Considerations

Leaving the discussion of whether or not you should backup user data out of the equation for a second, there is – in theory – little of Exchange's built-in features that really require a backup. Except for the

aforementioned exceptions that aren't stored in Active Directory, one could say that a backup of the configuration data (typically included in the system state) has become obsolete.

After all, a wise administrator would script the customizations that they apply to their servers so that they can be quickly applied for new server builds as well.

Backing up mailbox databases themselves is more obvious. Although an Exchange Server 2013 environment could be designed with no backups (or what Microsoft refers to as Native Data Protection), most companies still choose to use backups.

There are many good reasons why you need to keep a backup. For instance, it can be useful in case you would lose all your database copies – although that scenario would likely imply that you have bigger issues to deal with such as a rogue administrator.

Backups are also particularly useful when you have a specific data retention requirement which you cannot (or don't want to) fulfill using your current hardware. Whatever the reason is to create a backup is of little importance. The important part is that if you decide to use backups, that you do it correctly.

## Backup Software

There are many good products on the market which will allow you to backup/restore user data. Some utilities will also allow you to restore at the mailbox level or even single items from a mailbox. We will not cover the individual solutions here, but there are some general thoughts that apply regardless of the solution you choose.

### *"IP-less" Database Availability Groups*

IP-less Database Availability Groups, officially named Database Availability Groups without an Administrative Access Point can be a challenge to integrate with backup software.

Most backup utilities require an endpoint (the Administrative Access Point) to which they connect in order to create a backup of databases in a Database Availability Group. In an IP-less DAG, however, there is no endpoint to connect to. As a result many backup utilities do not (yet) support using these new DAG types.

Make sure to check with your Backup Software Vendor whether their solution supports this new feature before implementing it.

### *Windows Server Backup and the Renewed VSS Writer*

Although it comes for "free", Windows Server Backup is often forgotten as a valid backup utility, mainly because of a lack of advanced administration features.

For instance, Windows Server Backup cannot be ran remotely and must run locally on the server that you want to backup. There is no central management console which allows you to manage multiple

instances on different servers. More information on the capabilities of the Windows Server Backup tool can be found online<sup>19</sup>.

This being said, it does a very good job at taking backups and it is fully supported too. Like most of the solutions, Windows Server Backup will use the renewed Exchange VSS Writer to take its backups.

Like in Exchange 2010, the Exchange VSS Writer allows you to backup both active and passive copies. Before, the VSS write was part of the Information Store service. In Exchange 2013 the VSS Writer was rewritten and moved to the Exchange Replication Service. Despite this change, the Information Store service is still required too: you need make sure that both the Exchange Replication Service and Information Store service are started before you will be able to take a backup.

**Note:** Although you have many different backup types (Incremental, Differential, Full), Microsoft recommends only using full backups. One of the reasons being the log truncation which happens after a successful Full backup. Secondly, a full backup does not require additional log files that might have been lost during a server failure. This being said, if your last full backup dates from one day ago and you have no log files since the last backup, you might still lose data up until the last successful backup.

## Recovery Databases

When you want to selectively recover data from a database, you restore it into a Recovery Database. Recovery Databases are a way for Exchange to differentiate between a normal database and one that is used only for recovery purposes. This allows an administrator to have the normal database mounted while a recovered version of that database is used to retrieve specific data from it.

Recovery Databases have several limitations. For instance, it cannot be replicated between members of a Database Availability Group. Furthermore, users will not be able to “log on” to a Recovery Database.

Windows Server Backup doesn't natively support using Recovery Databases. When recovering a mailbox database using Windows Server Backup, you first need to restore the database to an alternate location after which you have to manually copy it into the Recovery Database's folder structure.

For more information see [Restore an Exchange Server 2013 Database to a Recovery Database<sup>20</sup>](#).

## Recovering Servers with the /m:RecoverServer Switch

The Exchange setup program includes a switch which is specifically designed to recover a failed server, **/m:recoverserver**.

<sup>19</sup> [http://technet.microsoft.com/en-us/library/dd876851\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/dd876851(v=exchg.150).aspx)

<sup>20</sup> <http://exchangeserverpro.com/restore-exchange-server-2013-database-recovery-database/>

When an admin invokes the /m:recoverserver switch, Exchange will be reinstalled onto the server using the configuration data stored in Active Directory. This will ensure that the server's configuration is nearly identical to its state prior to the failure.

**Note:** The Edge Transport Server cannot be recovered using the /m:recoverserver switch.

From a high-level perspective, a typical recover process includes the following steps:

1. Reinstall the operating system and join the 'new' server to the domain using the same name & IP address as the failed server
2. Run the Exchange setup program with the /m:recoverserver switch
3. Re-apply customizations and re-import certificates

Depending on what server roles your server was installed with, the recovery process might look a little different. In the step-by-step guides below, we will guide you through these specific scenarios.

**Warning:** When using the /m:recoverserver switch, it is recommended that you use the same version of Exchange Server 2013 that was previously installed on that server.

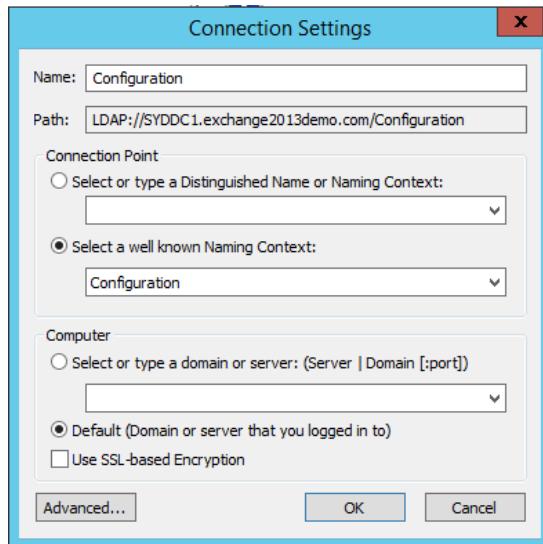
#### *Step 1: Verify Server Install Path*

Before starting the entire restore process, it is important that you verify the Exchange Server installation path of the failed server.

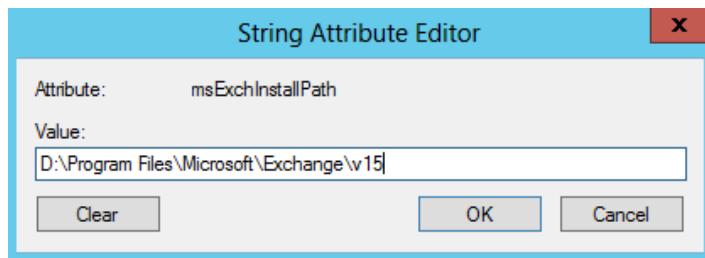
As long as the default location is used, you can skip this step. However, sometimes organization choose to deploy Exchange on a different drive or in a different folder.

Because the restored server needs to be configured in exact the same way, you must verify the current install path and specify that one with the /m:recoverserver switch if it deviates from the default.

1. Open **ADSIEDIT.msc** and connect to the **Configuration Naming Context**:



2. Navigate to the following location:  
**CN=<Exchange Server Name>,CN=Servers,CN=Exchange Administrative Group (FYDIBOHF23SPDLT),CN=Administrative Groups,CN=<Exchange Organization Name>,CN=Microsoft Exchange,CN=Services,CN=Configuration,DC=DomainName,CN=TLD**
3. Right-click the Server object and choose **Properties**
4. Locate the msExchangeInstallPath attribute and make a note of the value:



#### *Step 2: Reinstalling the Server and Joining it to the Domain*

The following steps assume you have already reinstalled the same operating system (Windows Server 2012 R2) on the replacement server.

**Warning:** If you decide to use new hardware for the replacement server, it is critically important that you ensure that the failed server is no longer connected to the network and that it will never be reconnected to the network again.

The first thing you need to do is reinstall the prerequisites<sup>21</sup> for Exchange 2013:

1. Open Windows PowerShell and execute the following commands. This will automatically install all the Operating System requirements.

```
[PS] C:\> Install-WindowsFeature RSAT-ADDS, AS-HTTP-Activation, Desktop-Experience, NET-Framework-45-Features, RPC-over-HTTP-proxy, RSAT-Clustering, RSAT-Clustering-CmdInterface, RSAT-Clustering-Mgmt, RSAT-Clustering-PowerShell, Web-Mgmt-Console, WAS-Process-Model, Web-Asp-Net45, Web-Basic-Auth, Web-Client-Auth, Web-Digest-Auth, Web-Dir-Browsing, Web-Dyn-Compression, Web-Http-Errors, Web-Http-Logging, Web-Http-Redirect, Web-Http-Tracing, Web-ISAPI-Ext, Web-ISAPI-Filter, Web-Lgcy-Mgmt-Console, Web-Metabase, Web-Mgmt-Console, Web-Mgmt-Service, Web-Net-Ext45, Web-Request-Monitor, Web-Server, Web-Stat-Compression, Web-Static-Content, Web-Windows-Auth, Web-WMI, Windows-Identity-Foundation
```

2. Once the above features have been installed, run the setup for the Microsoft Unified Communications Managed API 4.0<sup>22</sup>

After having performed these two steps, it's time to join the server to the domain. Because there was already a server with the same name previously registered, we need to reset the computer account first.

1. On a computer that has the Active Directory tools installed, open **Active Directory Users & Computers** and navigate to the Organization Unit where the failed server's computer account is located.
2. Right-click the computer account and select **Reset Account**.
3. When prompted to confirm the reset, click **Yes**.
4. Click **OK** in the confirmation prompt.

Next, move to the new server and join it to the domain using the same name as the failed server:

1. Open the Server Manager and navigate to **Local Server**.
2. Click the Computer Name to open up the **System Properties**.
3. Click **Change...** next to "*To rename this computer or change its domain or workgroup...*"
4. Enter the computer name and the domain and click **OK**
5. When prompted, enter domain credentials that have permissions to join a computer account to the domain.
6. Restart the server when asked to do so.

### *Step 3a: Recovering a Standalone Client Access, Mailbox or Multi-Role Server*

The following steps illustrate the recovery process of a standalone or multi-role server which is not part of a Database Availability Group.

<sup>21</sup> [http://technet.microsoft.com/en-us/library/bb691354\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/bb691354(v=exchg.150).aspx)

<sup>22</sup> <http://go.microsoft.com/fwlink/?LinkId=258269>

If that is the case the the recovery steps are as follows:

1. Open an elevated Command Prompt window, navigate to the location of the binaries and then type the following command.

```
C:\Admin\ex2013cu5>Setup.exe /m:recoverserver /IAcceptExchangeServerLicenseTerms
```

2. Once the process has completed successfully, reboot the server and continue to the post-recovery actions.

#### *Step 3b: Recovering a Mailbox Server in a Database Availability Group*

The outline of the process to recover a Mailbox Server that is part of a DAG is roughly the same as restoring a standalone Mailbox Server. The main difference lies in the fact that you need to remove the failed server from the DAG before it can be re-added. If you don't, the recovery process will fail with the following warning:

```
The Exchange server is a member of a database availability group.  
For more information, visit: http://technet.microsoft.com/library(EXCHG.150)/ms.exch.setupreadiness.MemberOfDatabaseAvailabilityGroup.aspx
```

1. First, start by verifying what database copies existed on the server and remove them.

```
[PS] C:\>Get-MailboxDatabaseCopyStatus -Server MELEX1 | Remove-MailboxDatabaseCopy
```

**Note:** Because the server hasn't been recovered yet, the above command will yield an expected error pointing out that it couldn't communicate with the failed server. This warning is expected and can safely be ignored.

2. Once all database copies have been removed, forcefully remove the failed server from the database availability group. The –ConfigurationOnly switch is used because the original server is not available so cannot be contacted to cleanly uninstall Exchange Server 2013 and remove configuration information from Active Directory.

```
[PS] C:\>Remove-DatabaseAvailabilityGroupServer DAG1 -MailboxServer MELEX1 -ConfigurationOnly
```

3. You are now ready to recover the server. Open an elevated **Command Prompt** window, navigate to the location of the binaries and then type the following command.

```
C:\Admin\ex2013cu5>Setup.exe /m:recoverserver /IAcceptExchangeServerLicenseTerms
```

After the process completed successfully, you will have to re-configure the server to be a member of the Database Availability Group:

4. Open the Exchange Management Shell and execute the following command.

```
[PS] C:\> [PS] C:\>Add-DatabaseAvailabilityGroupServer DAG1 -MailboxServer MELEX1
```

After the recovered server has successfully been added back into the Database Availability Group, you can reconfigure additional database copies.

**Note:** If this server previously hosted a lagged database copy, don't forget to reconfigure the same lag time when you are re-adding the database copies to the recovered server.

#### *Step 4: Post-Recovery Actions*

As mentioned earlier, you need to reconfigure a few items in order for the recovered server to be in the same state as before. Depending on what changes you have made, you will have to perform one or more tasks here.

One of the most important items to reconfigure are the URL settings of the different virtual directories. Although you could figure out what the configuration should be like by looking at your design- or as-built documentation, it's better to periodically take a snapshot of this information and store it in a safe place – especially after making configuration changes.

**Real World:** documenting the URL configuration can be a huge task, especially in larger environments. To avoid having to go through the process manually, Exchange MVP Michael Van Horenbeeck has created a script ([Get-VirDirInfo.ps1](#)<sup>23</sup>) which will query each Exchange Client Access Server for its URL configuration and then write it into a HTML report which you can backup more easily to another location.

Secondly, you need to re-import the certificate that was previously used. If you have multiple servers in your environment, you can easily export the certificate from another server and then re-import it on the recovered server.

---

<sup>23</sup> <http://vanhybrid.com/2014/02/12/exchange-virtual-directory-html-report/>

If you only have a single (Client Access) server, than you basically have two options:

1. Restore the certificate from a backup (e.g. an export you did earlier)
2. Re-key the existing certificate through the Certificate Authority by creating a new certificate signing request on the recovered server.

**Warning:** Although most Certificate Authorities will re-key a certificate for free, some do charge a moderate fee. Make sure that you take that and the time it takes to rekey a certificate into account.

It's not uncommon to see that a standalone Client Access server is used as the File Share Witness for a Database Availability Group. Not only is your Database Availability Group at risk while the File Share Witness is unavailable, you also need to re-confirm the recovered server as a File Share Witness. You can do this by temporarily moving the File Share Witness resource to another server and then move it back.

**Real World:** Sometimes when you are restoring a server which also served as a File Share Witness, you may end up in the situation where reassigning the File Share Witness might cause a problem with duplicate File Share Witnesses. Exchange Server MVP Michael Van Horenbeeck has [written an article](#)<sup>24</sup> about it which might be useful to read.

## Managing and Monitoring Summary

Managing and monitoring Exchange Server 2013 high availability environments requires a detailed understanding of the new Managed Availability features. Managed Availability can be both a blessing and a curse, in that it often takes necessary corrective actions and restores service automatically for you, but at the same time can be quite difficult to gain visibility of what Managed Availability is doing to your environment.

Exchange administrators also need to understand how high availability deployments impact the backup and recovery scenarios, particularly for servers that are members of a Database Availability Group.

---

<sup>24</sup> <http://vanhybrid.com/2014/07/15/spooky-the-curious-case-of-the-ghost-file-share-witness/>

# High Availability for Hybrid Deployments

A hybrid deployment is a term that refers to an IT system which spans both on-premises and cloud components. When we apply this logic to Exchange, a hybrid deployment is an organization's on-premises Exchange environment configured in such a way that it works together with that same organization's Office 365 Exchange Online tenant.

The configuration of a Hybrid Exchange Deployment is executed through the Hybrid Configuration Wizard (HCW) which was first introduced back in Exchange Server 2010 Service Pack 2. Before that, configuring a hybrid deployment was a tedious and cumbersome task which involved no less than 60 different configuration items.

In Exchange Server 2013 the HCW got a complete overhaul and it continues to be enhanced throughout the various Cumulative Updates that have been released so far. For example, Cumulative Update 5 for Exchange Server 2013 introduced the automatic configuration of OAuth.

For now let's focus on the hybrid architecture by taking a look at the components that come into play. This should make it easier to understand why certain components are required for high availability.

# Hybrid Concepts

Under the hood, the following four building blocks can be considered to be the cornerstones of a hybrid deployment:

- Secure Mail Flow
- Directory Synchronization
- Remote Mailbox Moves
- Rich Coexistence (Exchange Federation)

## Secure Mail flow

Mail flow between the on-premises organization and the online counterpart is – by default, when configured through the Hybrid Configuration Wizard – encrypted using TLS.

Because of the specific configuration, certain headers (such as the X-MS-Exchange-Organization-AuthAs header) are persisted when a message is sent across. This ensures that a message sent by an on-premises mailbox to an Office 365 recipient within the same organization is recognized as internal. If not, Outlook and Outlook Web App would recognize the message came from an external recipient and show a different behavior.

A good example that underlines the importance of this are Out of Office replies. If the X-MS-Exchange-Organization-AuthAs header (which controls this behavior) would not have the expected value (i.e. “internal” when receiving a message from a colleague who might be on-premises while your mailbox is in Office 365), colleagues would see the external OOF message instead of the internal one.

The mail flow component in a hybrid deployment is supported by both the Client Access- and Mailbox Server Role when you are not using an Edge Transport server in the perimeter network. The Client Access Server is the endpoint for incoming messages whereas outgoing messages are always sent from a Mailbox server. The outgoing messages can also be proxied through the Client Access server if the administrator chooses to enable that option.

If your deployment uses Edge Transport servers they will be used to route both incoming and outgoing messages. The TLS configuration is performed automatically by the Hybrid Configuration Wizard, and in Exchange Server 2013 also the HCW also takes care of the Edge Transport server configuration.

Secure mail flow requires you to have a trusted SSL certificate configured for the various transport components on both the Client Access and Mailbox Servers that you include in your Hybrid Deployment.

## Directory Synchronization

The Windows Azure Directory Sync Tool is commonly referred to as Directory Synchronization or even better known as just DirSync and it is a very important component in a Hybrid deployment.

DirSync synchronizes the attributes necessary to seamlessly onboard and off-board mailboxes to and from Office 365 by e.g. taking care of a mailbox' proxyAddress attribute. DirSync's primary purpose is to create a copy of your local Active Directory accounts in the Windows Azure Active Directory which fuels your Office 365 tenant. In addition, it will stamp each mailbox in the environment with an X500 address which will ensure that message flow is maintained even when a mailbox is moved cross premises.

DirSync is also responsible for providing a "unified Global Address List" experience and it enables the use of features such as Exchange Online Archiving (an on-premises mailbox with In-Place Archive hosted in the cloud) by enabling the write-back into Active Directory of certain attributes.

**Note:** The experience of a Global Address List is only upheld if no Address Book Policies are created on-premises.

While DirSync is a critical component to ensure that a hybrid deployment is working correctly, an organization can easily survive without DirSync for a few hours. However, during that time no changes can be made to recipients and no mailboxes can/should be moved. But more about that later.

## Remote Mailbox Moves

As you might be aware, the Mailbox Replication Service is the component which enables an administrator to transparently move mailboxes between databases within an Exchange deployment. By leveraging the same components for mailbox moves on-premises, the same experience can be guaranteed when moving mailboxes cross-premises.

In order to be able to use the Mailbox Replication Service, Exchange Online communicates with it through the Mailbox Replication Service Proxy (MRS Proxy). By default the MRS Proxy is disabled, but is enabled as part of the Hybrid Configuration Wizard in Cumulative Update 5 and later.

The MRS Proxy lives on the Client Access Servers and is only used when mailboxes are moved between the on-premises organization and Exchange Online. As such, whether or not you require high availability for this feature depends on how critical mailbox moves are to your organization.

When an organization is only setting up a Hybrid deployment with the purpose to move mailboxes to Office 365, then high availability for the Mailbox Moves is often deemed non-essential. After all, Hybrid is often only a bridge to the cloud, not a permanent solution. The downside is that there might be delays with on-boarding to Office 365 if the single Hybrid server is offline.

## Rich Coexistence (Exchange Federation)

Exchange Federation isn't new. It was first introduced in Exchange 2010 and was a welcome alternative to allow organizations to share Free/Busy information without having to go to great lengths to make that work. In order to exchange Free/Busy information with another organization, one has to setup a

trust with the Microsoft Federation Gateway and a trust with the remote organization called an “Organization Relationship”.

The Microsoft Federation Gateway (MFG) is a free service, provided by Microsoft, which acts as a security broker between two different Exchange organizations.

By having a trust with the MFG, organizations are no longer required to exchange credentials with one another. When a request is made from Organization A to Organization B, Organization A would first contact the MFG to request a token. This token would then be sent along with the request for Free/Busy information to Organization B.

Before honoring this request, Organization B would then verify with the MFG whether or not the token that Organization A has sent is valid. If it is valid then Organization A is proved to be trustworthy and the Free/Busy information is returned. Of course, a request would only be possible if Organization A had an Organization Relationship with Organization B and vice versa.

In a Hybrid deployment the two organizations are your on-premises organization and the Exchange Online tenant.

This method of federation is still available today in Exchange 2013. However, some organizations (and countries) have opposed to using a 3<sup>rd</sup> party authentication broker. While the MFG is a secure service, some organizations do not allow offloading authentication to a third party.

As such, Microsoft introduced OAuth – an industry standard protocol already used in other products such as Lync and SharePoint 2013. Exchange 2013 allows you to setup a direct trust with a remote organization using OAuth instead of the Microsoft Federation Gateway. If you then setup an Intra-Organization Connector or IOC, you have a new method of exchanging Free/Busy between organizations without having to rely on the MFG.

For now, this scenario is only supported in a hybrid deployment and an on-premises organization that uses Exchange 2013 SP1 or later. The Hybrid Configuration Wizard will setup both Federation and OAuth ever since Cumulative Update 5 for Exchange Server 2013, as long as you no longer any Exchange Server 2010 or older servers in the environment. If you still have legacy servers, the option to automatically configure OAuth will not be shown in the HCW and you will have to configure it manually by following a series of steps, documented on TechNet<sup>25</sup>.

In a mixed on-premises deployment where both Exchange 2010 and 2013 is used, it might very well be that both methods are used interchangeably. It all depends on who generated the Free/Busy request and where that person’s mailbox is located.

If a user’s mailbox is located on an Exchange 2013 Mailbox Server, OAuth will be used. If the user’s mailbox is located on a legacy Exchange server, the Federation Gateway will be used. Of course, that is if both methods are configured and working. Unfortunately, Exchange will not attempt to do a “fail back”

---

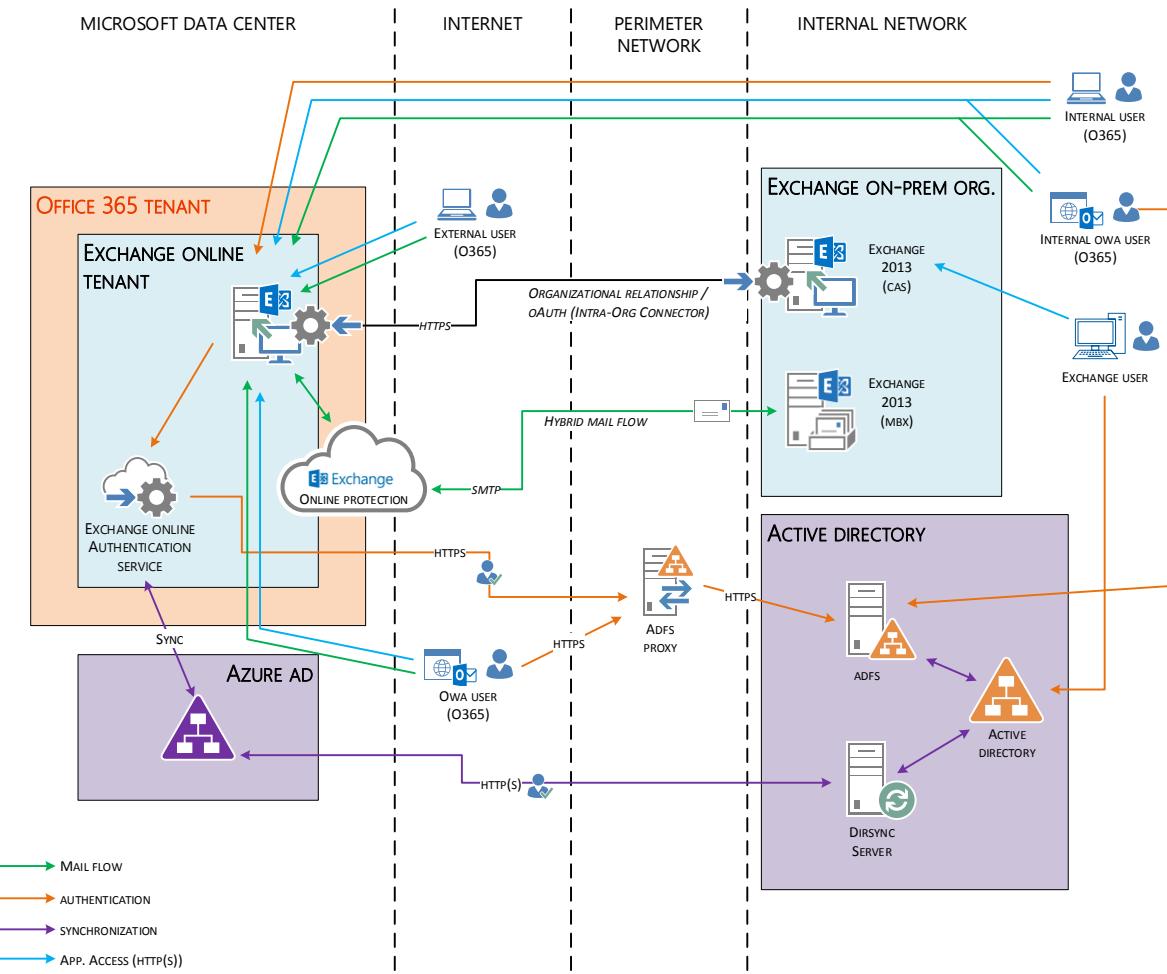
<sup>25</sup> [http://technet.microsoft.com/en-us/library/dn594521\(v=exchg.150\).aspx](http://technet.microsoft.com/en-us/library/dn594521(v=exchg.150).aspx)

to using the Microsoft Federation Gateway if requesting Free/Busy using OAuth does not work. Hence, it's critical to keep an eye on these features too.

## Bringing it all Together

The following diagram depicts the different components that come into play in a Hybrid Deployment. It should provide you with a better understanding of where the components we just discussed are being used. As you can see a Hybrid Deployment is quite complex.

Additionally, the diagram also contains more information on the different client connections:



## Why is High Availability Important in a Hybrid Deployment?

There are various reasons why an organization chooses to deploy a hybrid configuration. As mentioned earlier, some organization just use it as a temporary measure to allow them to move mailboxes from their on-premises organization to Office 365.

In such case, you might not be bothered with exchanging Free/Busy requests or setting up high availability for the Hybrid components.

The purpose of this chapter is not to explain how you can selectively configure certain aspects of a hybrid deployment, instead it will provide you with information on how to setup high availability for the “bigger picture”.

The question as to why high availability is required is basically the same as why an organization chooses to deploy high availability on-premises. If you are willing to accept having limited functionality in case a server is having problems, than not taking additional actions is fine.

However, a Hybrid deployment is much more than only Exchange. There are many more components at stake than just an Exchange Server. Aside from Exchange and DirSync there is also the question of how to handle authentication.

Many customers choose to deploy a hybrid configuration supported by Active Directory Federation Services (AD FS). AD FS gives an organization the ability to provide Single Sign On to their end-users. As such, users no longer need a separate user account and password to access their cloud-based services in Office 365.

Choosing to deploy AD FS means that you are accepting the responsibility of being responsible for authentication of users accessing Office 365 resources. If your AD FS servers are unavailable then your users will not be able to access their mailbox in Exchange Online.

Hence, if you choose to use AD FS it's critical that you consider deploying this role highly available. In fact, I would go as far as saying that it's mandatory if you want to ensure continuous service availability.

As an alternative to AD FS, more and more organization are looking into a feature called “Password Synchronization” which is a sub-feature of Directory Synchronization. As the name of the feature might give away, it will synchronize a user's on-premises password to Office 365. More specifically, it will synchronize the hash of the password's hash. Therefore, the user's unencrypted password or the encrypted password itself never leaves your local environment.

Passwords need to be synchronized only once; at least that is until they are changed. When you are using Password Synchronization, it is somewhat less critical to have this feature setup highly available, as long as you keep the following in mind - when Password Sync is unavailable, password changes won't be synchronized to Exchange Online. This could potentially cause some confusion, but at least people can still access their mailboxes. Usually the password for Office 365 is saved in Outlook. So, even if the password changed on-premises, Outlook would not re-prompt to enter the password.

If we cycle back to Transport again, remember that customers can handle mail flow in a hybrid scenario in two ways - either all messages are routed through the on-premises organization or they leverage Exchange Online Protection to handle all inbound mail traffic.

Outbound, organizations can still choose to send message directly to the internet from Exchange Online or through the on-premises Exchange Servers. Regardless of the solution that is opted for, mail flow is a critical feature which must be considered for high availability –even in the smallest deployments.

## Other High Availability Components in a Hybrid Deployment

Having all the high availability options in the world for any of the aforementioned components can be great; it will do you no good if you have no reliable or redundant (internet) connection.

All access either by Exchange or clients is over the Internet; it's not possible to connect your organization's network to the multi-tenant Office 365 environment. The loss of internet connectivity means that your users will not be able to connect to Office 365. And if you are using AD FS for authentication, this also means that users outside of your organization might not be able to connect to servers in your organization, which might prevent them from authenticating (depending on your setup).

The result is that configuring high availability for a hybrid deployment is about much more than just adding a second Exchange, DirSync or AD FS server. It's about ensuring connectivity and functionality regardless of the type of failure. In addition to just considering your internet connectivity, if your organization uses proxy server infrastructure, ensure that this is either bypassed or is both sized for Office 365 traffic and provides the same level of high-availability.

## High Availability for Directory Synchronization

Let's start with the easiest component, DirSync. As mentioned earlier, there is no real need to have a highly available DirSync setup. Even more so, there is no way of doing it - at least not built into the product.

By default DirSync will use a locally installed SQL Express database where it will store its Metaverse. If you are familiar with Forefront Identity Manager, the Metaverse is a single view of the world –a staging area where objects are written to before they are synchronized to the destination (Azure AD, in this case). This staging area is then used in subsequent synchronizations to determine which objects and which object attributes have changed. Only these changes will then be synchronized, greatly reducing the time needed to perform full synchronizations.

If that database is stored locally, you risk losing the data in the Metaverse when the disk crashes. Although that is not necessarily a problem, because you can deploy a new DirSync server which will then automatically create its own Metaverse to work with. However depending on the size of your organization that process – which happens during the initial synchronization -- could take up quite a while; several hours for larger environments.

If you are not concerned about how long it takes to restore the DirSync functionality, this might not be a concern. Many customers, from the smallest to organizations with tens of thousands of users find the simplicity of this approach outweighs the cost of deploying a standby DirSync server.

But if you want to avoid this delay, you can also configure DirSync to use a database hosted on a separate SQL server. For larger environments with over 50,000 synchronized objects this is required anyway, because a SQL Express database is limited in size and might not be sufficiently large to meet the space requirements for large environments.

As an alternative, many customers deploy a standby DirSync server. Often, this is a virtual machine which sits idle and waiting for its services to be started, or for DirSync to be installed by an Administrator in case of a failure of the primary DirSync server.

This can be particularly useful in a scenario where you rely on Password Synchronization and you want to limit the time that it is unavailable when the primary server goes down. If you choose this approach, it's important to ensure that whenever you upgrade the DirSync version on the primary server you also make sure that same version is available on your standby server.

## High Availability for Client Access Servers in a Hybrid Deployment

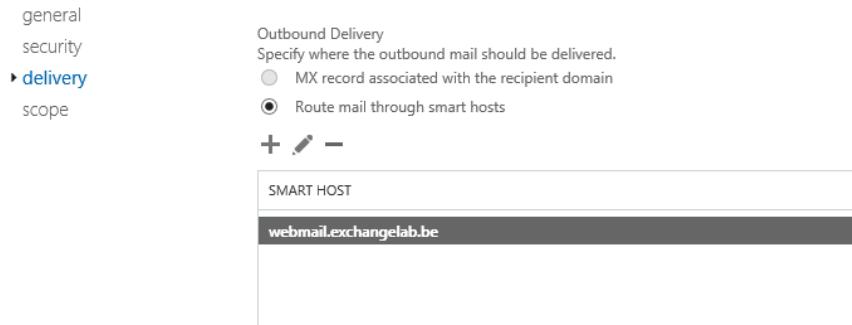
As outlined earlier, the Client Access Server fulfills several tasks such as incoming mail, incoming Free/Busy requests and remote mailbox moves.

Earlier in this guide we discussed high availability for the Client Access server role. If we apply the same HA logic here, adding high availability to the Client Access components in a hybrid deployment is identical to how you would do it for an on-premises only deployment – by using multiple load-balanced Client Access servers. This also means that you can just use your existing highly-available CAS deployment.

Typically, the different components in a hybrid deployment will use the same external namespace(s) as your on-premises organization. By leveraging Autodiscover, Exchange Online will try to find out to what URLs it needs to connect to for Free/Busy requests and remote mailbox moves. For some other features, like mail flow the endpoint (URL) is hard-coded in the configuration.

The following screenshot displays the hard-coded on-premises FQDN for mail flow from Office 365 to Exchange on-premises:

Outbound to 970418a1-dab1-4b9b-b453-9bd8ec642732



For inbound mail flow, you might be tempted to point the URL to an on-premises mail filtering solution like Cisco IronPort or Barracuda Spam Firewall. Unfortunately, this would strip certain SMTP headers from messages that are sent between both environments and is therefore not supported.

If you need a solution which can be deployed in the perimeter network and is highly available, you will have to look at deploying multiple load-balanced Edge Transport servers, or simply using a load balancer in the perimeter network as a layer 4 load balancer or “proxy”.

## High Availability for Mailbox Servers in a Hybrid Deployment

If you need high availability for outbound mail flow, than you will need at least two on-premises mailbox servers. Again the deployment principle is no different from deploying multiple mailbox servers for a purely on-premises deployment. Only now, there is no need to deploy a database availability group to ensure high availability for outbound (or inbound) mail flow, as mail flow is handled by the Transport services.

Of course, if you already have highly available Mailbox servers on-premises, there is no specific need to add separate “hybrid” mailbox servers; you can just as easily use the existing servers and add them to your hybrid configuration.

From a configuration point-of view, you need to ensure that the same certificate is assigned to the transport components on each mailbox server. If not, the certificate will not be available to choose from in the Hybrid Configuration Wizard.

The following image depicts the transport certificate configuration during as part of the Hybrid Configuration Wizard:

## Modify Exchange Hybrid

Choose a certificate to use with securing hybrid mail transport.  
[Learn more](#)

outlook.exchangelab.be

outlook.exchangelab.be

## Regional Considerations for Mailbox Migrations

Before diving into regional deployment considerations, let's first have a look at how mailbox moves occur in a hybrid deployment.

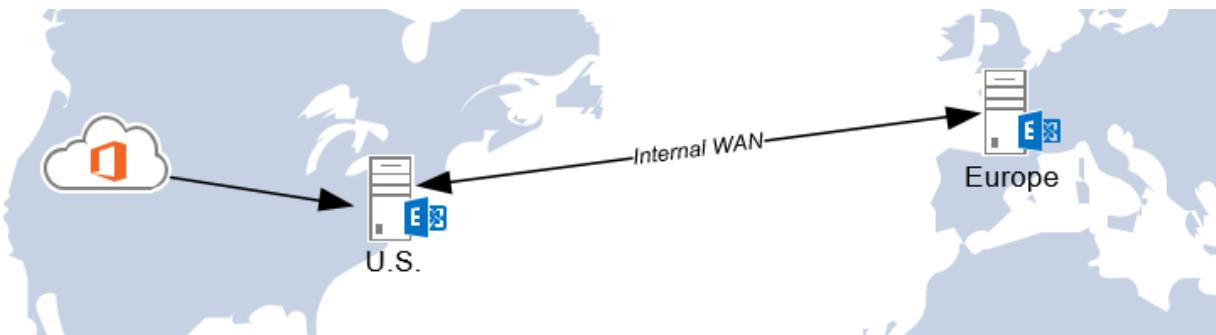
When a move request is generated, Exchange Online will make an inbound connection to the on-premises Exchange environment. The endpoint that it connects to is referred to as the migration endpoint. Using Exchange Online PowerShell, you can query which endpoints have been created using the Get-MigrationEndpoint cmdlet:

```
PS C:\> Get-MigrationEndpoint
```

Identity	EndpointType	RemoteServer
-----	-----	-----
webmail.exchangelab.be	ExchangeRemoteMove	webmail.exchangelab.be

By default, the first migration endpoint is generated when the first mailbox move is performed. The endpoint URL will be based on what is returned by the Autodiscover service.

If your on-premises organization spans multiple sites or multiple geographical locations, you might not always want to use the same endpoint to move data. This because doing so might cause data to travel over the internal WAN instead of the internet as depicted in the image below:

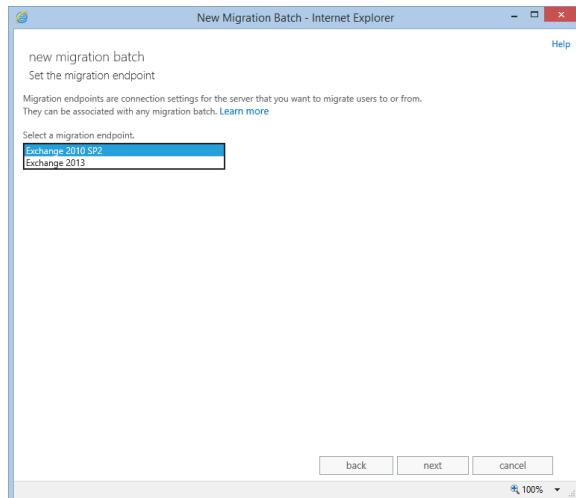


As such, it's the organization which carries the cost for moving data from Europe to the Office 365 tenant, instead of leveraging relatively cheap internet bandwidth.

To circumvent this problem, an administrator can create multiple migration endpoints by using the New-MigrationEndpoint cmdlet.

```
PS C:\> New-MigrationEndpoint -RemoteServer webmail.exchange.lab.be -ExchangeRemoteMove -Name Europe
```

Once you have created additional endpoints, you will have the option to choose from one of the endpoints when creating a new migration batch:



Similar to this approach, you could define different namespaces for the same geographical location. This is often done in case a hybrid deployment is only used for performing mailbox moves and the hybrid deployment needs to intervene as little as possible with daily operations.

For example, your organization has an on-premises Exchange 2010 deployment and decides to deploy additional Exchange 2013 servers for hybrid purposes. Normally when you are planning to host mailboxes on Exchange 2013, you would have to deploy Exchange 2013, point all Client Access traffic to it and then setup a hybrid deployment.

If you perform these tasks for Hybrid, the effort for doing this is quite similar to a full on-premises migration. Some organizations want to avoid this additional work and decide to setup a separate namespace (e.g. hybrid.domain.com) which then points only to the Exchange 2013 servers.

All other namespaces which existed previously continue to point to the existing Exchange 2010 environment, except for Autodiscover which must be switched over to Exchange 2013 anyhow. This is because many features in a hybrid deployment rely on Autodiscover.

Once you have setup the hybrid deployment this way, a new migration endpoint (which points to Exchange 2013 instead of Exchange 2010) can be created and mailbox moves can be performed through there. Of course, this example might be a little silly as you can use Exchange 2010 SP3 to setup a Hybrid

deployment without having to deploy Exchange 2013. But you could use the same approach just as easily with an on-premises Exchange 2007 environment and hybrid Exchange 2013 servers.

## High Availability for Transport in a Hybrid Deployment

High availability for outbound mail flow is configured semi-automatically. Once you have multiple Exchange 2013 servers in your environment, it suffices to add those to the Hybrid Configuration Wizard to allow them to send messages to Exchange Online –provided that you have configured all those servers with the correct SMTP certificate.

Inbound mail flow (from Office 365 to Exchange on-premises) is a little different. If you have a single site, ensuring that your inbound route is highly available and there are multiple servers on the receiving end should be sufficient. But what about when the primary ingress point fails? For instance, due to an internet connection outage?

The solution is as simple as it is elegant. Exchange Server MVP, MCSM and transport expert Brian Reid provides [guidance on how to perform this configuration on his website](#)<sup>26</sup>.

When the Hybrid Configuration Wizard configures mail flow between Office 365 and Exchange on-premises, it will configure an Outbound Connector in Exchange Online with a smart host that points to the on-premises environment. Typically, this smart host is represented by an A record in the Public DNS zone which then points to a single on-premises environment. But if you replace that A-record with multiple MX records, you effectively leverage SMTP's built-in capability to fail over to a different host if the first one is unavailable. This mechanism is described in [RFC 2821](#)<sup>27</sup>.

As such, instead of treating the smart host as a single DNS record, it is treated like a zone name with multiple MX records. Imagine that the smart host you configured is called “smarthost.domain.com”, you could then create the following MX records:

Record Type	Preference	Destination
MX	10	Europe-in.smarthost.domain.com
MX	10	USA-in.smarthost.domain.com
MX	20	Europe-DR.smarthost.domain.com
MX	20	USA-DR.smarthost.domain.com

In this example only the MX records with a preference of 10 will be used for inbound mail flow. If these endpoints become unavailable, the other two hosts (with a preference of 20) will be used. As you can

<sup>26</sup> <http://www.c7solutions.com/2014/03/highly-available-office-365-to-on-premises-mail-routing>

<sup>27</sup> <https://www.ietf.org/rfc/rfc2821.txt>

derive from the names that we used, this approach is also valid to create geo-redundancy for inbound mail flow.

**Note** For this approach to work, you must make sure that your public DNS zone only contains MX records for the smart host that is specified in the Outbound Connector in Exchange Online.

## Authentication Using Active Directory Federation Services

High availability for AD FS is pretty straightforward. By deploying multiple AD FS servers and joining them to the same AD FS farm, you can scale out your authentication infrastructure. Similar to the Client Access Server infrastructure, it suffices to add a load balancer to divide the load and determine when a server is unhealthy.

There are several ways to determine the health of an AD FS server. Prior to AD FS in Windows Server 2012 R2, you had lots of IIS components that could be used for that. Since Windows Server 2012 R2, however, AD FS does no longer require you to deploy IIS for it to work. As such, it becomes a little more challenging to determine its health.

The panacea would be to execute a synthetic transaction against each server in the array which determines whether or not it was able to request a SAML token. That would require the load balancer to support this functionality, which not many vendors offer today.

Alternatively you could check the availability of the federation metadata page. If the load balancer can load the page, you can safely assume that the AD FS server is still working. If the Active Directory Federation Services are not running on the server, the page won't be loaded either. Even though it's not a watertight solution, it sure beats doing a simple ping to see whether the server is alive or not.

In August 2014, Microsoft came out with an update for AD FS in Windows Server 2012 R2. This update introduces a HTTP-based Health Check Probe functionality<sup>28</sup> for the AD FS Server Role and the AD FS Server Proxy which is now part of the Web Application Proxy (WAP).

This feature allows a load balancer to probe a specific URL (i.e. `http://<Web Application Proxy name>/adfs/probe`) to verify the health of the AD FS Proxy server. Of course, this is a far better and more integrated way of determining the health of either. As such, I would highly recommend deploying this update so you can benefit from this improvement.

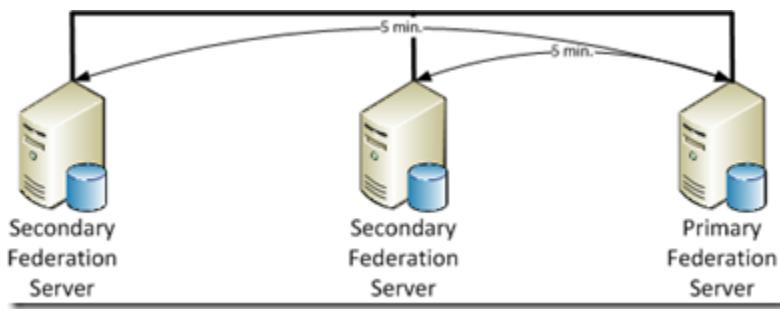
---

<sup>28</sup> <http://blogs.technet.com/b/applicationproxyblog/archive/2014/10/17/hardware-load-balancer-health-checks-and-web-application-proxy-ad-fs-2012-r2.aspx>

There are several ways how to create an AD FS server farm and the way you do will influence how HA operates. This has mainly to do with the choice of the database for AD FS. By default, when you install Active Directory Federation Services, it will use the Windows Internal Database.

Surprisingly, the Windows Internal Database option is actually a solid and easy way to deploy AD FS, even if you have a few servers deployed throughout the farm. In general, you can assume that you can easily deploy a 4 server AD FS farm using the built-in database. The maximum amount of servers in a farm using the Windows Internal Database is 5.

When you do use the Windows Internal Database, the first server you install into the farm will be designated as the “primary” federation server. As you join other servers to the farm, these will periodically pull the farm’s configuration information from the primary AD FS server:



By default, this happens every 5 minutes, is fully transparent and doesn’t require the admin to do or setup anything. When the primary AD FS server fails, the other servers in the farm won’t be able to pull information from it which will prevent you from making any change to the AD FS farm’s configuration. Obviously, this is not a good thing, but it’s not the worst that could happen either. After all, each AD FS server in the farm operates independently and continue to do so even after your primary AD FS server may have failed.

If you cannot restore the server for some reason, then you can promote a secondary AD FS server to become the primary using the Set-ADFSyncProperties cmdlet.

```
[PS] C:\>Set-ADFSyncProperties -Role PrimaryComputer
```

**Note:** This command must be run from the AD FS server which you are designating as the new primary server.

Once you have configured a new Primary Federation Server, you must reconfigure the remaining secondary servers to point to the new primary server using the Set-ADFSyncProperties cmdlet.

```
[PS] C:\> Set-AD FSsyncProperties -Role SecondaryComputer -PrimaryComputerName <fqdn of primary server>
```

Despite this limitation, I personally prefer using the Windows Internal Database approach because it's much easier and faster to deploy. Nonetheless, using the built-in database is not an option for every organization. Some are too large (greater than 50,000 users) or might have specific security requirements which requires to use one of the features described below. These features only work when a dedicated SQL database is used in the farm:

- SAML Token Artifact resolution
- SAML/WS-Federation Token Replay Detection

Another pain-point in AD FS server farms that use the built-in database is the requirement to be in the same time zone. Organization that want to span a farm across multiple locations world-wide, will have to configure remote servers to be in the same time zone as the primary AD FS server. If they don't do that, the replication mechanism between the primary and secondary servers will fail.

Of course, you can see that when you decide to use a dedicated SQL database, the need for HA for a SQL Server database also comes into play.

## Considerations for Publishing AD FS onto the Internet

It is not recommended to publish AD FS servers directly onto the internet, not even just through a load balancer – unless that load balancer has a built-in AD FS proxy or other security features. Usually though, if your load balancer does not provide this option, you would need some sort of highly available reverse proxy solution which can handle AD FS traffic.

In Server 2008 R2, the AD FS proxy server role was a simple to deploy and lightweight secure proxy, dedicated to AD FS traffic. A typical implementation included one or more AD FS Proxy servers which are configured to proxy traffic for the AD FS server farm and then load balance request across all these proxy servers. Unlike AD FS servers, there is no need to setup a farm for the AD FS Proxy servers – all that you need to do is configure the AD FS Proxy servers to talk to the underlying AD FS server farm.

In Windows Server 2012 R2, the AD FS Proxy role has been replaced by the Web Application Proxy. They both operate in a similar way and configuring high availability is also done in the same way, by adding a load-balancer in front of them. The Web Application Proxy includes additional useful features, including secure application publishing including the ability to use publish Exchange and other applications using AD FS.

Alternatively, there are several third party solutions which can replace the AD FS or AD FS Proxy server. Some load balancer vendors (such as F5 or Citrix) have this capability. If you use one of these devices, it

might be beneficial to use that feature; it could save you from having to deploy a bunch of AD FS Proxy servers. And if these devices are setup with HA, you're all set without too much additional effort. If you decide to purchase or use a 3<sup>rd</sup> party federation solution, make sure it is certified as "Works with Office 365"<sup>29</sup>.

## Hybrid AD FS Deployments

Active Directory Federation Services can itself be deployed in a hybrid configuration. This hybrid configuration has nothing to do with Exchange though. It doesn't require a drastic reconfiguration of AD FS, nor does it change how high availability for AD FS works.

Basically, you host one (or more) AD FS servers at a different location such as a cloud infrastructure provider such as Microsoft Azure or Amazon Web Services. That way, if your on-premises AD FS servers are unavailable, the cloud-hosted AD FS servers can take over.

The way this works is that authentication requests are balanced across both AD FS servers. The cloud-based AD FS server will then authenticate the request by talking to an on-premises Domain Controller over the VPN which ties both environments together.

Provided that this VPN is highly available, there is little risk here. But if your VPN would fail, the cloud-based AD FS server would no longer be able to service requests as it cannot talk to a Domain Controller anymore.

The same thing would happen if the on-premises environment would fail (e.g. datacenter outage); only then things get worse as both AD FS servers would fail, beating the purpose of having one AD FS server in Azure.

Hence, the recommendation to deploy at least a single domain controller along with the AD FS server in Azure. As such, you can guarantee operations independently of the VPN connection or the state of each of the locations:

In this scenario, we need one more component to make this solution water tight - a load balancer which is able to detect if the on-premises or Azure-based AD FS server is available or not. However, because doing so isn't particularly easy and could end up costing quite a bit, some organization choose to setup their AD FS infrastructure this way, but keep the Azure-based AD FS server as a hot standby (or the other way around).

This means that in case of a failure of the on-premises AD FS server, a manual interaction is required to change the destination of the A record to point to the Azure-based AD FS server instead of the on-premises one.

This introduces a short period of time during which new authentication request cannot be serviced, but it will probably be a lot easier to manage; not to mention less expensive.

---

<sup>29</sup> <http://blogs.office.com/2013/09/03/works-with-office-365-identity-program/>

# Authentication with Password Synchronization

Even though many of the organizations that deploy a hybrid Exchange configuration choose to use AD FS, there are some where Password Synchronization is used instead.

It's actually a pretty good alternative and is very lightweight compared to AD FS. At present, for most implementations, the user experience accessing Exchange Online is identical, especially when using the Outlook client which doesn't support SSO yet. There are scenarios where it isn't same, such as web-based applications that use Windows Integrated Authentication to allow seamless access from domain joined PCs. These clients will need to sign in with a username and password instead.

Because Password Synchronization is part of DirSync, the same principles for high availability apply. The only catch here is that DirSync by itself isn't really critical. It does not need to be recovered immediately when it goes down.

Password Synchronization on the other hand shouldn't be unavailable for an extended period of time. When Password Synchronization is first enabled, it will do a full sweep of the on-premises passwords to Office 365. This means that all the users' passwords in Office 365 will be overwritten by the one that is used on-premises - provided these user accounts are no federated identities that use AD FS instead.

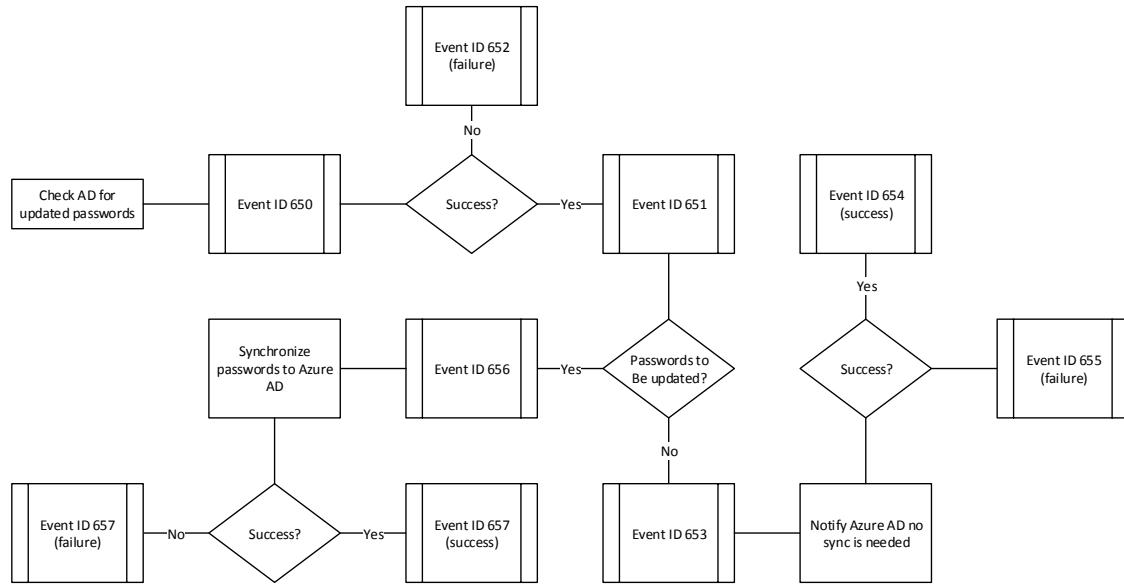
Each time a user changes his password on-premises, Password Synchronization will pick up that change and sync the new password to Office 365. After the initial synchronization of the password hashes, the Password Synchronization feature will check for updated passwords every two minutes. This means that passwords that are changed on-premises should be synchronized to Azure AD within a few minutes.

Now, imagine that Password Synchronization is down for some reason. Let's say the DirSync server crashed somehow. That would mean that everything continues to work as before. However, when a user changes his password on-premises, that password won't be synchronized to Office 365. When that user then tries to access Office 365 services with his new password, they obviously will get a password error.

Until password synchronization is re-enabled and a full sweep is forced the old password would still have to be used for all Office 365-based services. While services are still accessible, it's definitely going to cause some confusion – not to mention that it could potentially generate a bunch of calls to the help desk.

In order to keep track of what Password Synchronization is doing, it's recommended to take a look at the Application Event Log on the DirSync server. When the feature is checking the on-premises Active Directory, an event ID 650 will be written to the event log, followed by either an event ID 651 which depicts that the feature was able to retrieve passwords from the Active Directory or an event ID 652 telling the opposite (failure).

When Password Synchronization detects there are passwords that need to be synchronized to Azure AD, it will log an event ID 656. This event will be followed by an event ID 657 mentioning the success or failure of the synchronization. If no on-premises password changes are detected, the feature will log an event ID 653 to the event log, followed by either an event ID 654 (success) or event ID 655 (failure) to notify Azure AD there are no password synchronizations needed.



Because Password Synchronization is a feature of the DirSync tool, its high availability is directly dependent on DirSync, which we discussed earlier in this chapter.

## Hybrid Deployment Summary

As you can see hybrid deployments are complex environments with many elements that need to work together for the overall solution to function correctly.

You also saw that a variety of decision points exist for how a hybrid deployment will be created, from choosing between different version of Exchange Server, different authentication models, and also mail flow between the on-premises organization and the Exchange Online tenant.

Fortunately much of the hybrid deployment can leverage existing investments in on-premises high availability, such as multiple servers being deployed and load balancing.

# Appendix A – Lab Guide

The best way to learn about Exchange Server 2013 is to build your own test lab environment. A simple test lab environment gives you the freedom to experiment with different technical designs, explore features, and experience a wide range of failure conditions so that you can learn the right and wrong way to respond.

With powerful consumer PC hardware being available at a reasonable price, virtualization technology, and free trial editions of software it is within the reach of many IT professionals to have some level of test lab capacity available to them for their own personal use.

Exchange Server MVP Jeff Guillet regularly explores the availability of low cost hardware when he is expanding or replacing his own test lab. Most recently he published the details of his "[4<sup>th</sup> Generation Hyper-V 2012 R2 Server](#)" which cost around \$1200 USD<sup>30</sup>. Check it out if you're interested in building one for yourself.

Fortunately, for those that don't have the budget to build a server like Jeff's, there are other options as well. A regular PC or laptop with enough memory (RAM) and a fast hard drive (preferably SSD) can use the Hyper-V built in to Windows 8.1 to host a test lab environment.

In fact, as little as 16Gb of memory may be enough to run a test lab that consists of a domain controller, two Exchange Server 2013 servers, and possibly even a Windows 7 client machine. That configuration will allow you to explore most of the features of Exchange Server 2013.

Of course, you're welcome to use any hardware and virtualization platform that you prefer for your own test lab.

---

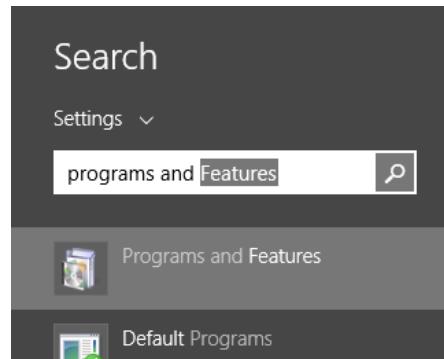
<sup>30</sup> <http://www.expta.com/2013/11/4th-generation-hyper-v-2012-r2-server.html>

# Preparing to Build a Test Lab Environment

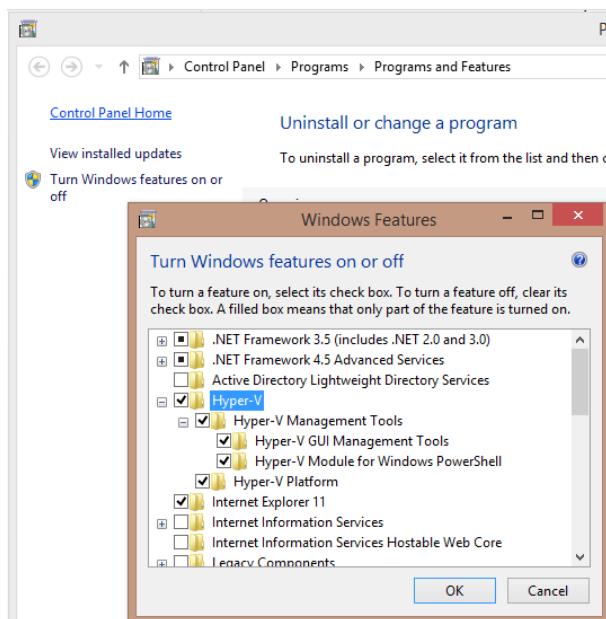
First let's take a look at the preparation steps for setting up a test lab environment using Hyper-V on Windows 8.1

## Installing Hyper-V on Windows 8.1

Although Windows 8.1 ships with Hyper-V as a feature it is not enabled by default. To enable Hyper-V first open Programs and Features.



Select **Turn Windows Features On or Off** and make sure that Hyper-V and all sub-components are ticked.



If Windows prompts you to restart after enabling these then go ahead and reboot your computer. While you are rebooting you should also go into your system BIOS and confirm that virtualization support is

enabled. This may appear under one of a variety of names such as “Intel VT” or “Hardware-assisted Virtualization Technology”. Refer to your motherboard manual if you need more guidance.

## Downloading Software

To build a test lab environment using the latest software available at the time of writing this guide you should download the following from Microsoft.

- Windows Server 2012 R2 – this will be an ISO file
- Exchange Server 2013 Cumulative Update 6 – you don’t need to look for Exchange Server 2013 and Cumulative Update 6 as separate downloads. All Exchange Server 2013 Service Pack or Cumulative Updates are a full build of the product.
- Windows 7 with Service Pack 1, or Windows 8.1 – some people prefer Windows 7 over Windows 8.1 so feel free to choose either.

Newer versions of this software may be available by the time you are reading this, so you should double check whether there is a more recent build available.

Generally speaking you can just download the latest version available at the time, however there may be minor changes that make things slightly different for you than what is demonstrated in this guide.

## What Else Do You Need?

In addition to a host machine and the software for installing your virtual machines you should also consider purchasing your own domain name if you do not already own one.

Owning your own domain name means that you can establish real inbound/outbound mail flow between your test lab environment and the rest of the world.

Obviously you should not try to use the exact same domain name that is used as a demonstration in this guide.

However if you accidentally create your Active Directory with the same DNS name as ours, or a DNS name that is not valid on the internet (eg “domain.local”), you can still register your own domain name to be used for email addresses in your Exchange Server 2013 organization.

## Building Virtual Machines in Hyper-V

To save on disk space you can build your virtual machines by first creating a base image that is then used as the source disk for other virtual machines. This is achieved using a Hyper-V feature called Differencing Disks.

If you prefer to just install each virtual machine independently without using Differencing Disks, but this will use more disk space on your host machine.

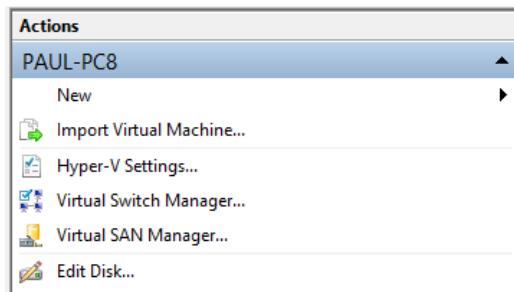
In addition to using Differencing Disks we can also use a dynamically sized disk for the base image, so that a reasonable system volume size can be allocated (eg 120Gb) without actually consuming all of that disk space on your host machine.

We know that 120Gb might sound like a lot of disk space just for a test server, but Exchange Server 2013 expects to see a large system volume so it is better to give it one. Use the combination of Differencing Disks and Dynamically Expanding disks reduce your overall disk space usage.

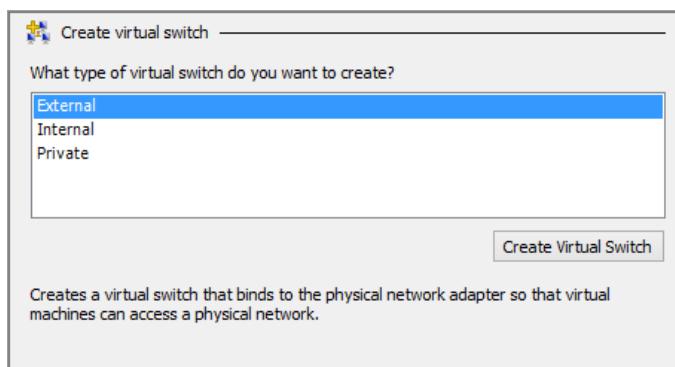
## Configuring a Virtual Switch

If this is the first time you've used Hyper-V on your host machine you'll need to configure a virtual switch for the virtual machines to connect to.

Open Hyper-V Manager and click **Virtual Switch Manager**.

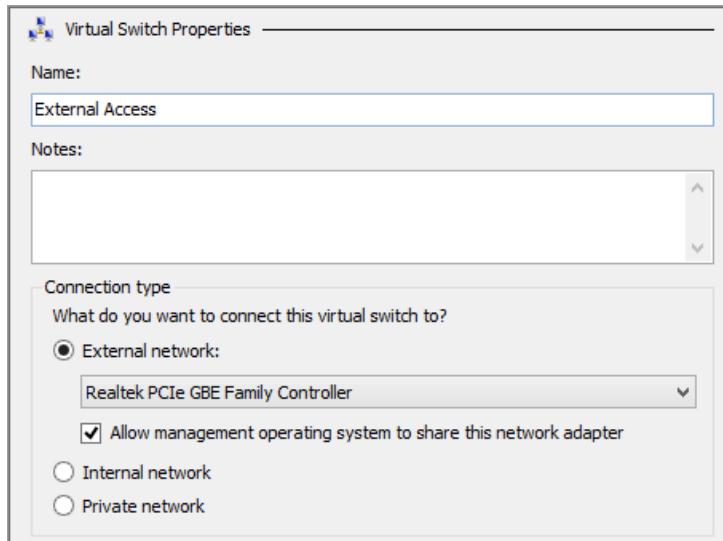


Choose **External** as the type of virtual switch, and click **Create Virtual Switch**.



Give your virtual switch a meaningful name such as "External Access", and configure it to use a network adapter in your host machine that is connected to your local area network.

If you only have one network adapter you can share it with the management operating system (your host machine's operating system) as well.

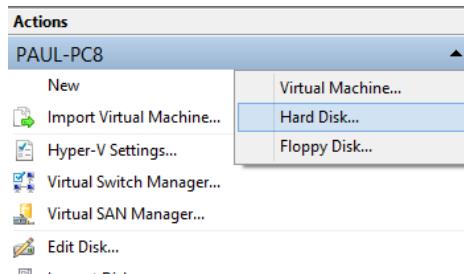


Click **OK** or **Apply** to finish creating the new virtual switch.

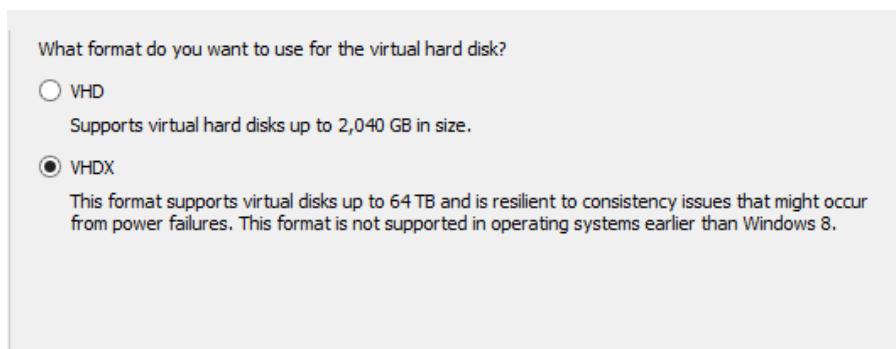
## Creating a Virtual Hard Disk to Use as a Base Image

Before you create the new virtual machine itself you should first create the virtual hard disk.

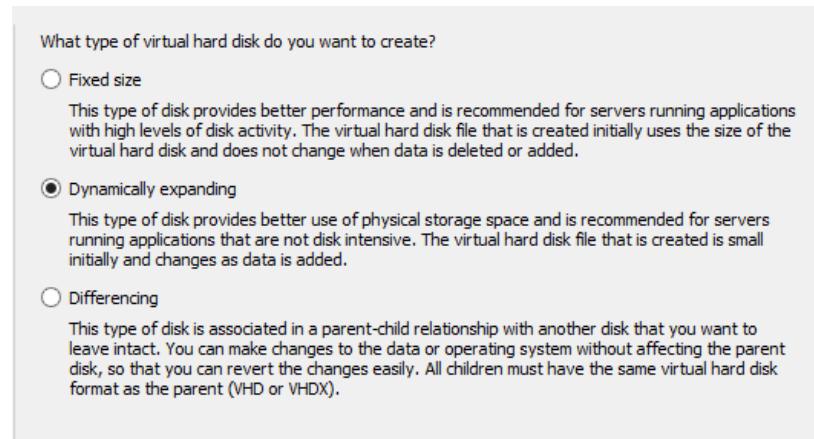
Open Hyper-V Manager and click **New → Hard Disk**.



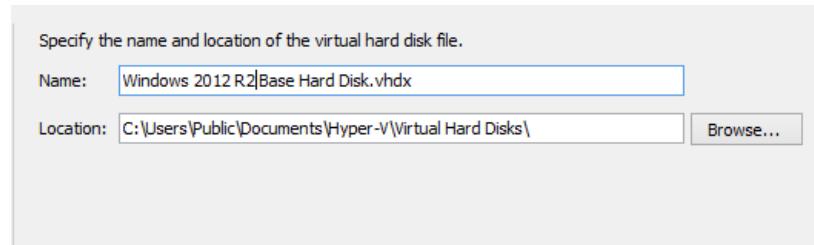
Skip past the Before You Begin page. Choose the **VHDX** format for your virtual hard disk and click **Next** to continue.



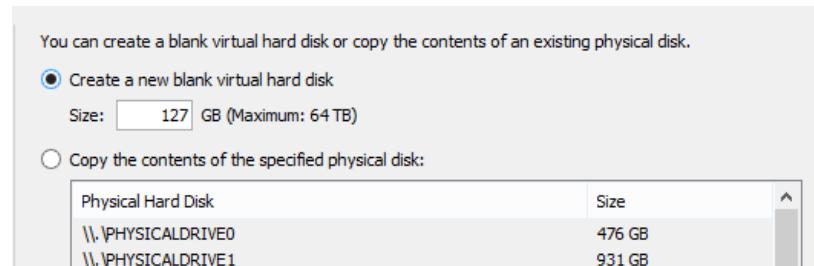
Choose the disk type of **Dynamically Expanding**, and click **Next** to continue.



Give the virtual hard disk a meaningful name such as “Window 2012 R2 Base Hard Disk” and choose a location for the file to be placed. Click **Next** to continue.



The default size of 127Gb is fine for our usage. Click **Next** to continue.

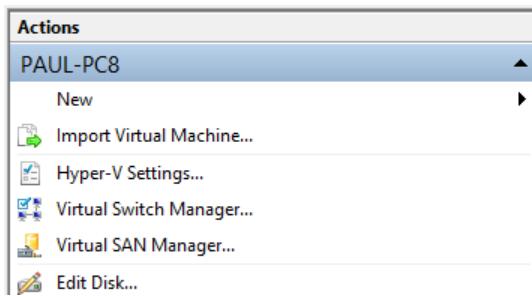


Double-check your choices on the summary page and if you’re happy with everything click **Finish** to create the virtual hard disk.

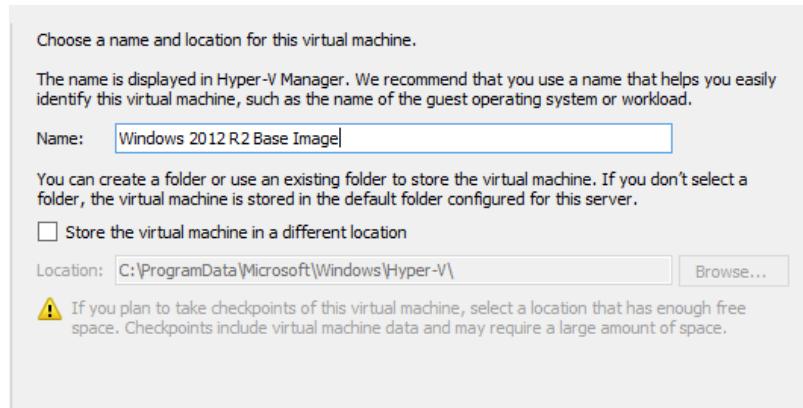
# Creating a Virtual Machine to Use as a Base Image

Now that we've configured your virtual switch and a virtual hard disk for your base image we can go ahead and install the virtual machine that will be used as a base image for all of our other Windows Server 2012 R2 virtual machines.

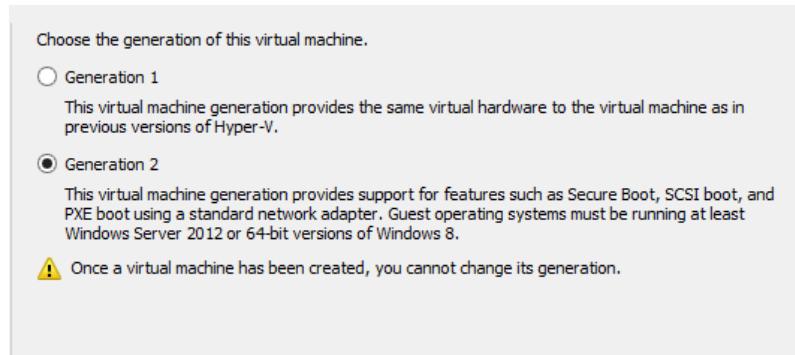
Open Hyper-V Manager and click **New → Virtual Machine** to create a new virtual machine.



Skip past the Before You Begin page, and give your new virtual machine a name. Because this is eventually going to be used as a base image we've called this one "Windows 2012 R2 Base Image". Click **Next** to continue.

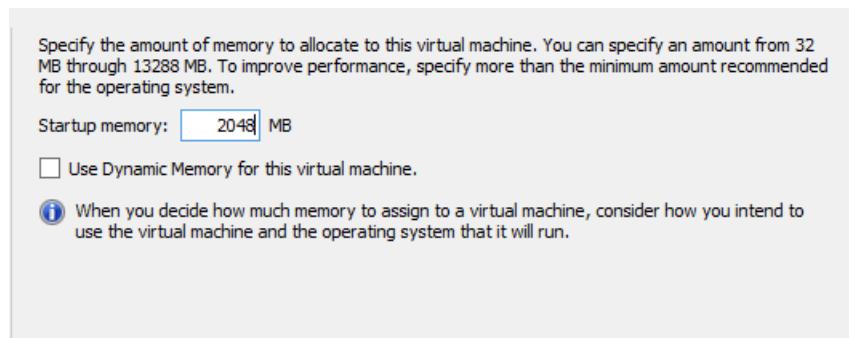


Although Generation 1 virtual machines will work just fine, we can choose Generation 2 as we are installing a compatible guest operating system. Click **Next** to continue.

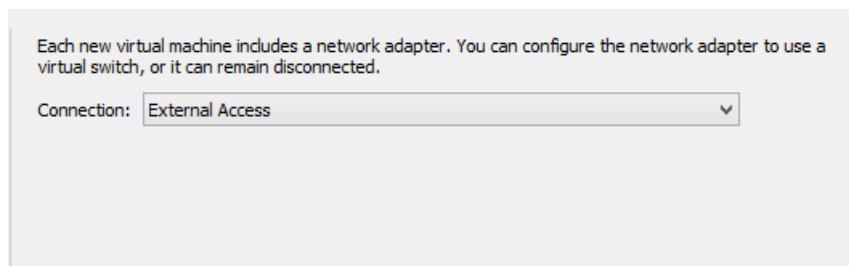


Assign enough memory so that your virtual machine will perform well while you are configuring it in readiness to become your base image.

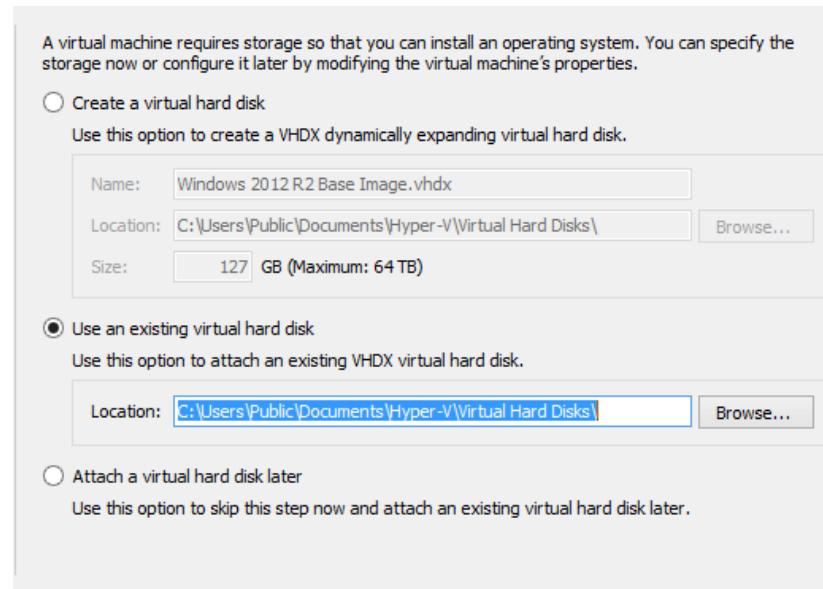
We've assigned 2Gb in this case. Click **Next** to continue.



Select the virtual switch that you configured to allow your virtual machines to connect to your network and the internet. Click **Next** to continue.



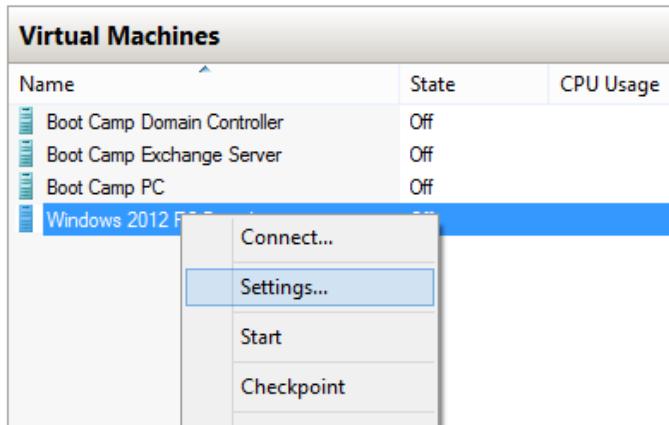
Select **Use an existing virtual hard disk** and browse to the location where your VHDX file was created.



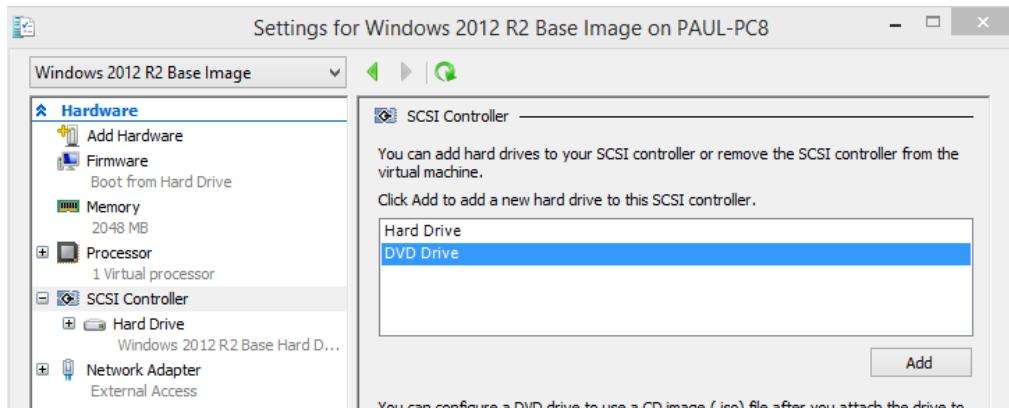
After selecting the file click **Next** to continue.

Review your selections on the summary page and click **Finish** to create the virtual machine.

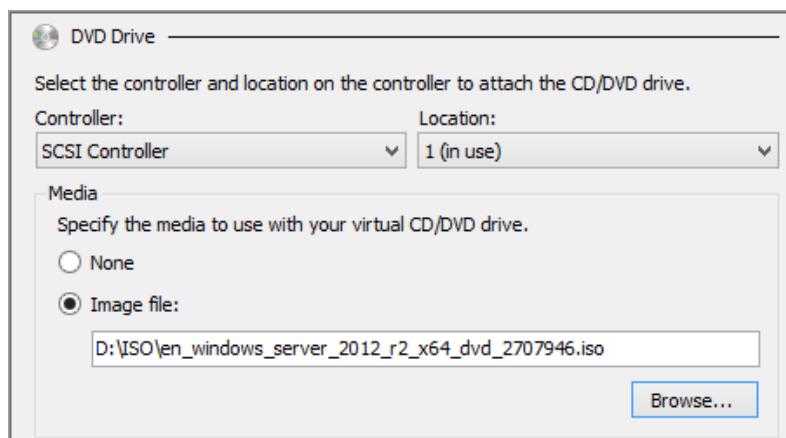
After creating the virtual machine we need to connect the ISO file that contains the Windows Server 2012 R2 installation media. In Hyper-V Manager right-click the virtual machine and choose **Settings**.



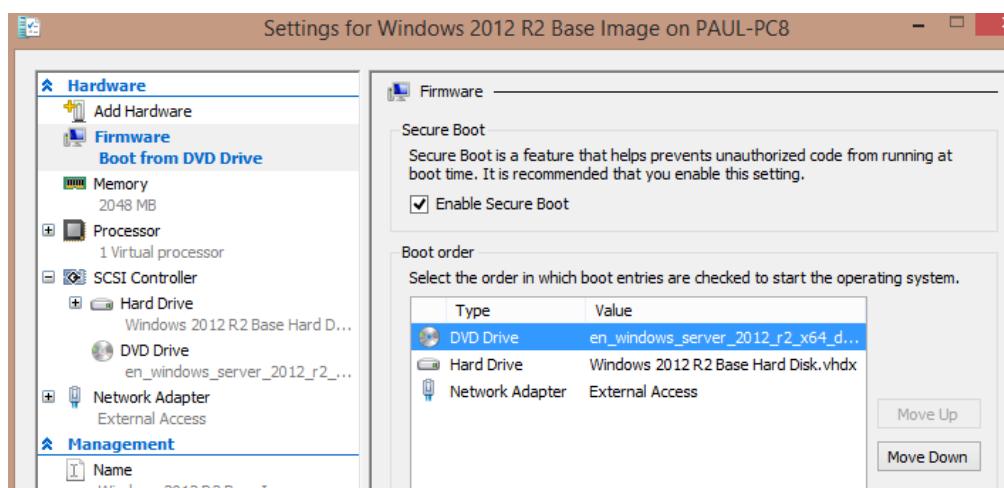
Select the virtual SCSI Controller and add a DVD drive.



Click **Browse** and select the ISO file that you have downloaded for Windows Server 2012 R2.

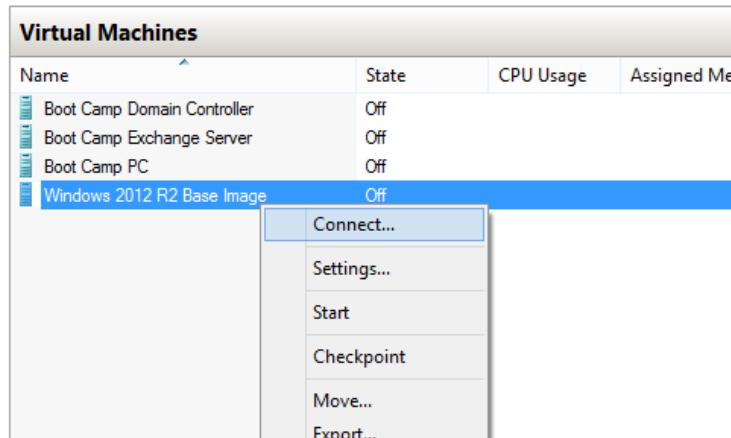


Click **Apply** to apply the hardware changes. Next, select the Firmware settings and move the DVD Drive to the top of the boot order.

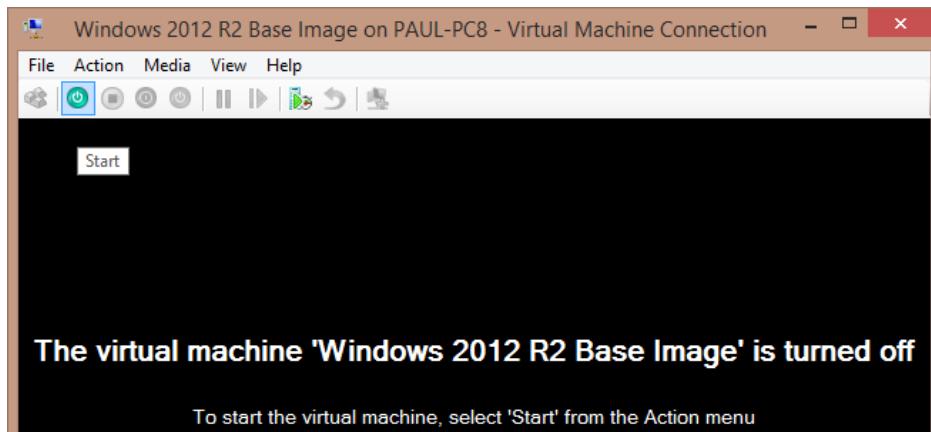


Click **OK** to apply the changes and close the Settings window.

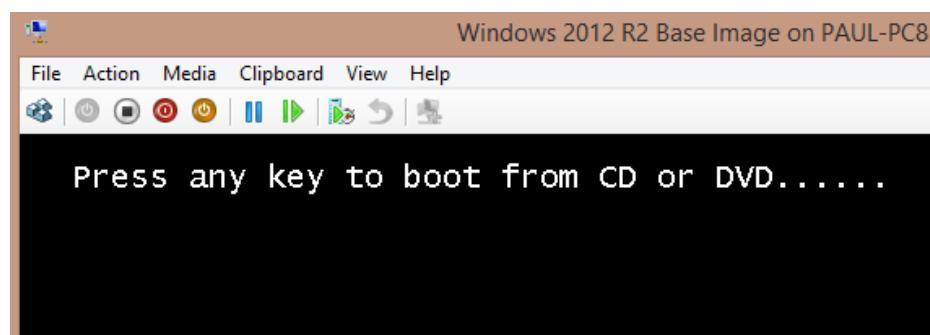
Connect to the virtual machine so that you can see the console.



Click the power icon to start the virtual machine.



At the prompt press any key to boot from the DVD drive.

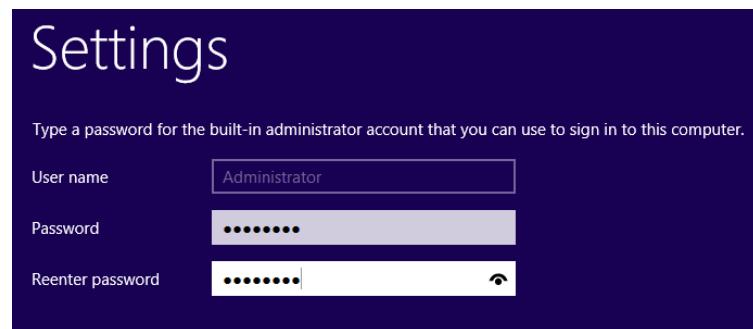


Windows Server 2012 R2 setup will begin.



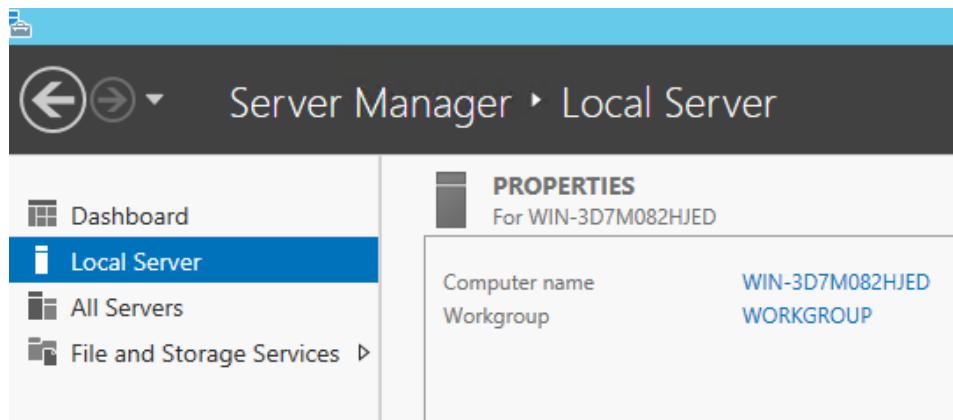
There is very little customization to perform during this stage of setup. Simply follow the setup prompts choosing the appropriate language and regional settings for your country, enter your product key (one is issued to you if you download the trial software), and make sure you choose to install **"Server with a GUI"** not "Server Core Installation".

On first boot Windows will prompt you to choose an administrator password.



After Windows has finished booting log in using the administrator password you chose.

The Server Manager console will automatically launch. Choose **Local Server** from the left hand side.

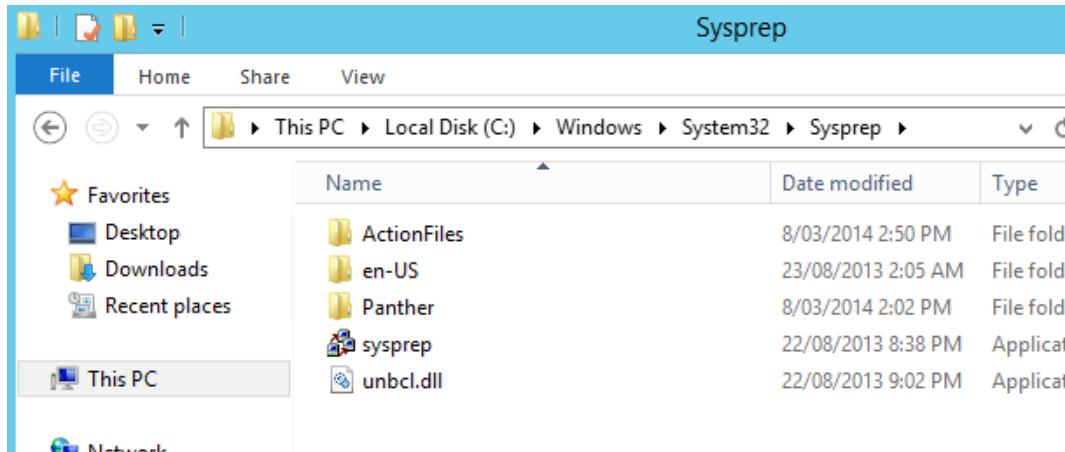


Because this virtual machine will become a base image for other virtual machines there is only a small amount of customization that we want to do at this stage.

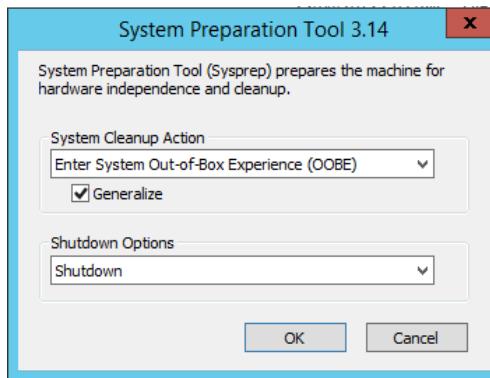
- Configure Windows Update and run an update to install any patches that are available
- Set the time zone to suit your location
- Turn off IE Enhanced Security Configuration
- Enable Remote Desktop

After finishing those initial configurations and restarting for Windows updates to finish installing we can Sysprep the server.

The Sysprep files are located in **C:\Windows\System32\Sysprep**.

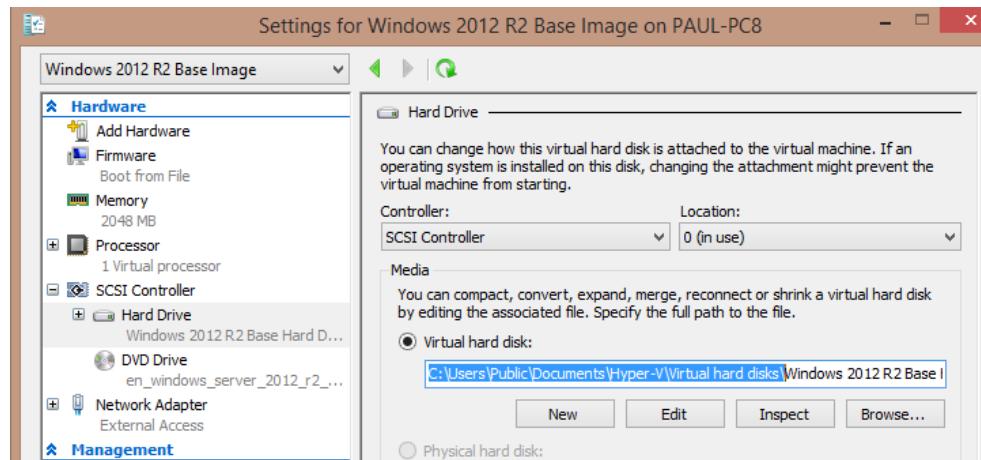


Run **Sysprep** and tick the box to **Generalize**, and set the **Shutdown Options** to “Shutdown”.

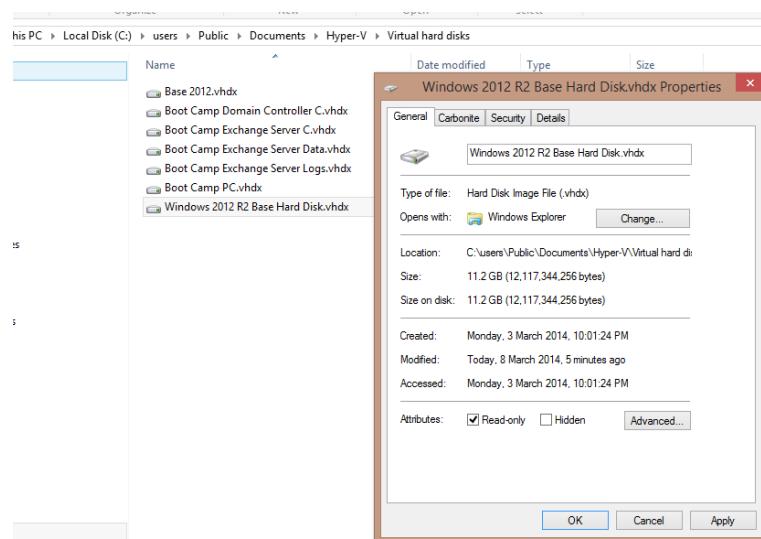


Click **OK** and wait a few minutes for the virtual machine to shut down.

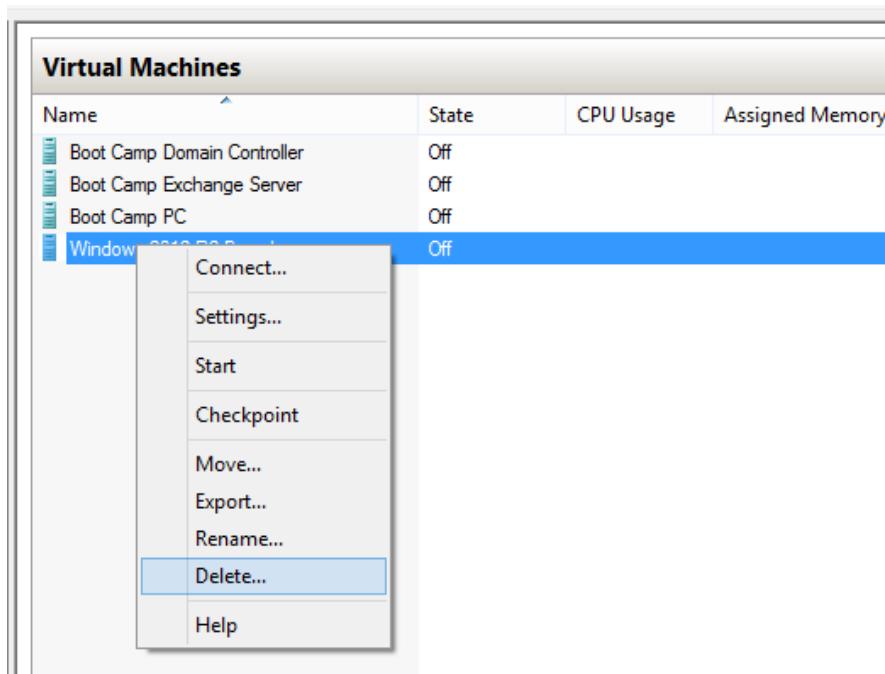
Open the virtual machine settings again and look for the location of the virtual hard drive.



Navigate to the location in Windows Explorer and set the file to **Read Only**.



In Hyper-V Manager we can also delete the virtual machine itself. This will leave the virtual hard drive on your computer to be used as the base image for other virtual machines.

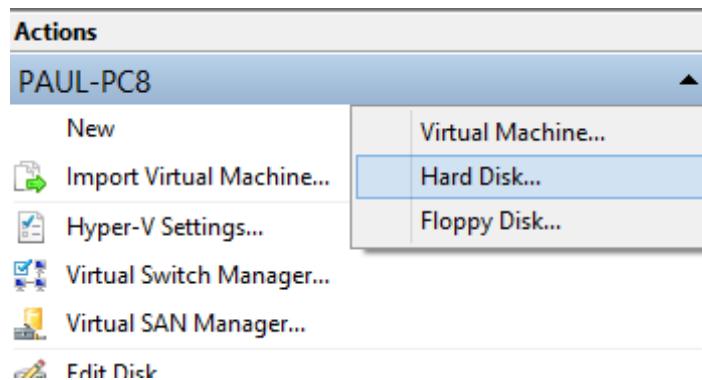


Now we can begin building virtual machines to use in the test lab environment

## Installing the Domain Controller

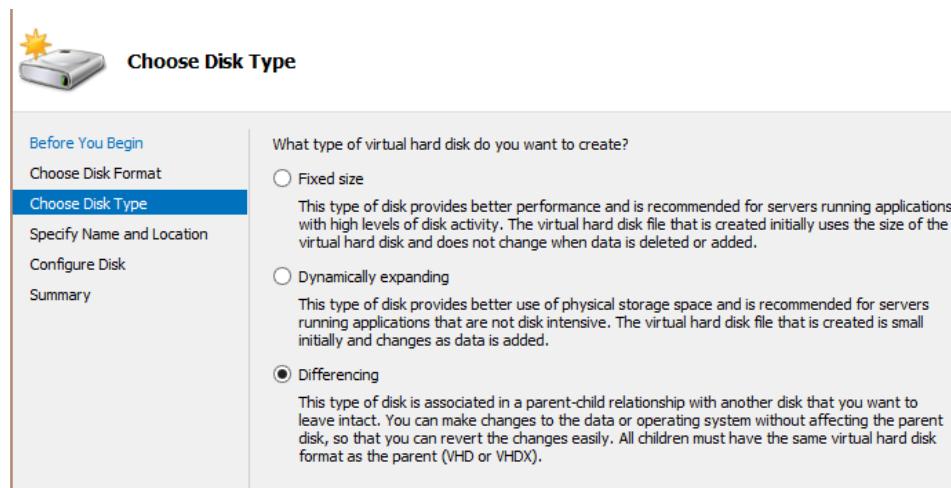
To begin the setup of a virtual machine to be the Active Directory domain controller we first need to create the differencing disk for the virtual machine.

In Hyper-V Manager select **New → Hard Disk**.

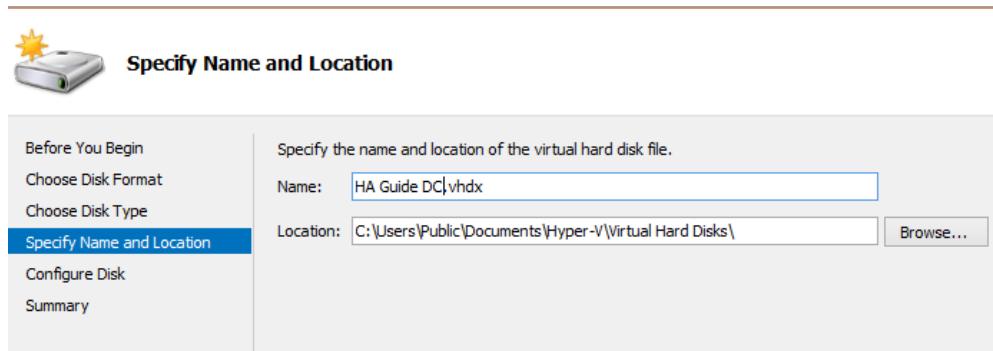


Create a new hard disk in VHDX format.

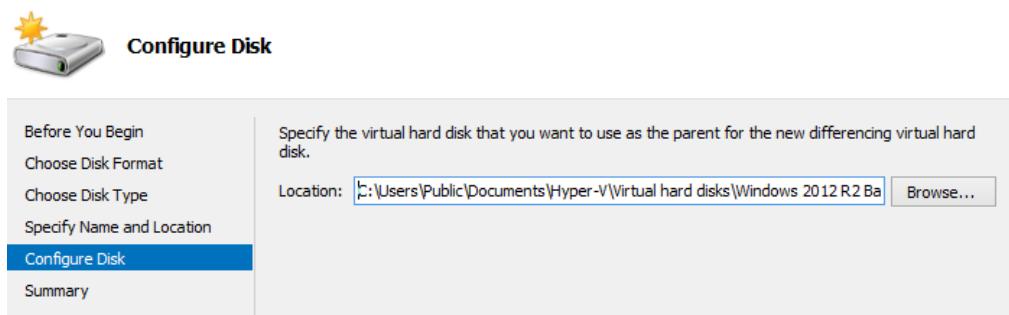
Choose the Disk Type of **Differencing**.



Give the hard disk a meaningful file name and choose a location to store it.

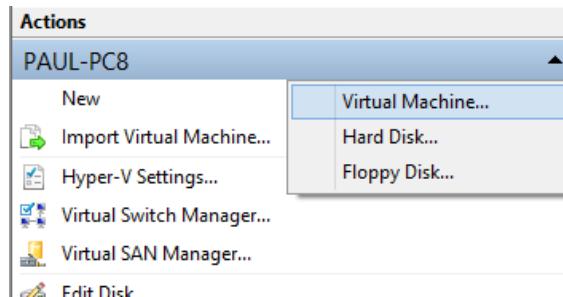


For the parent drive select the virtual hard drive that you created as a base image in the previous section of this chapter.



On the Summary page verify your selections then click **Finish** to create the virtual hard drive.

Next, create a new virtual machine.



Follow the same steps to create the virtual machine as you used in the previous section of this chapter.

- Give the virtual machine a unique, meaningful name such as “HA Guide DC” or “Test Lab DC”
- Give the virtual machine at least 2Gb of memory if possible
- Connect the virtual machine to your virtual switch
- When connecting a virtual hard disk choose to use an existing virtual hard disk and select the differencing disk you created earlier in this section

**Tip:** If you have a host machine with 16Gb of memory you might run into situations later where one of your Exchange servers won’t start in Hyper-V due to insufficient available memory. One of the ways you can work around this is to configure your domain controller virtual machine to use Dynamic Memory, and set a lower values for minimum and startup memory amount, such as 512Mb minimum and 1024mb startup, with a maximum memory of 2048Mb.

After creating the virtual machine we can power it on and enter regional settings, a product key, accept the license, and enter a local administrator password.

When we look at Server Manager we can see that the base image configurations for Remote Desktop, IE Enhanced Security Configuration, and the Time Zone are already set the same for the new server.

So we just need to:

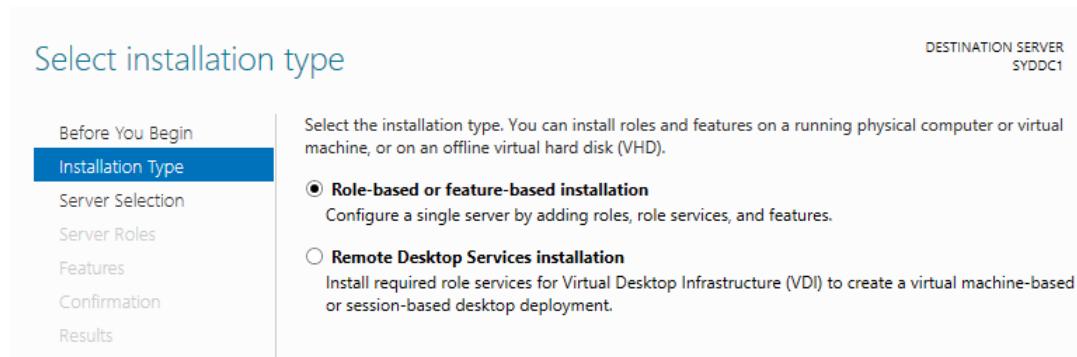
- Enable Windows Update again
- Configure a static IP address
- Configure a computer name

The computer name change will require a restart.

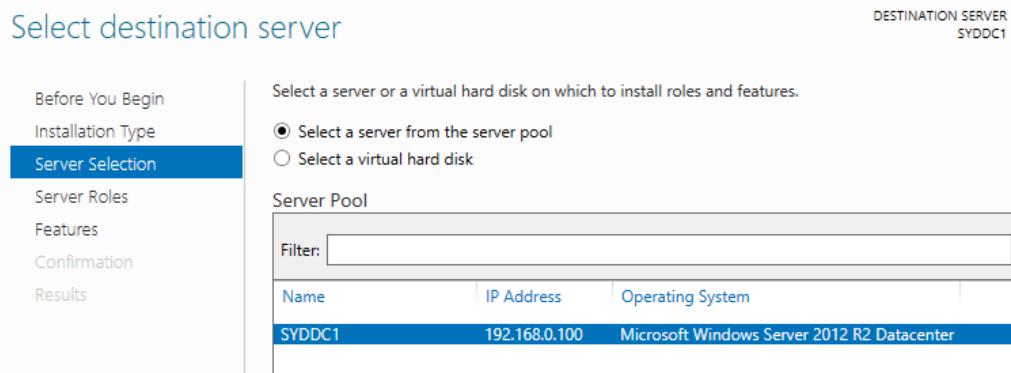
After restarting we can log back in and go to Server Manager again.

In the **Manage** menu select **Add Roles and Features**.

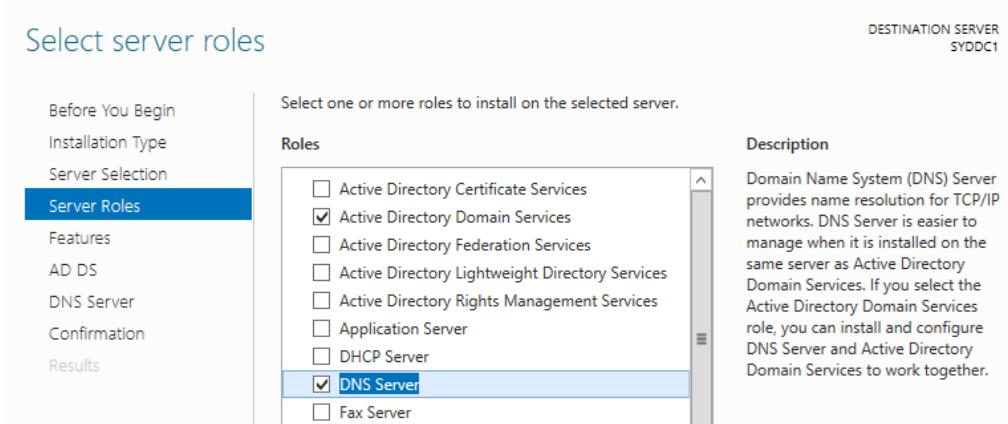
Skip past the Before You Begin page and choose **Role-based or feature-based installation**. Click **Next** to continue.



Select the server from the server pool. Click **Next** to continue.

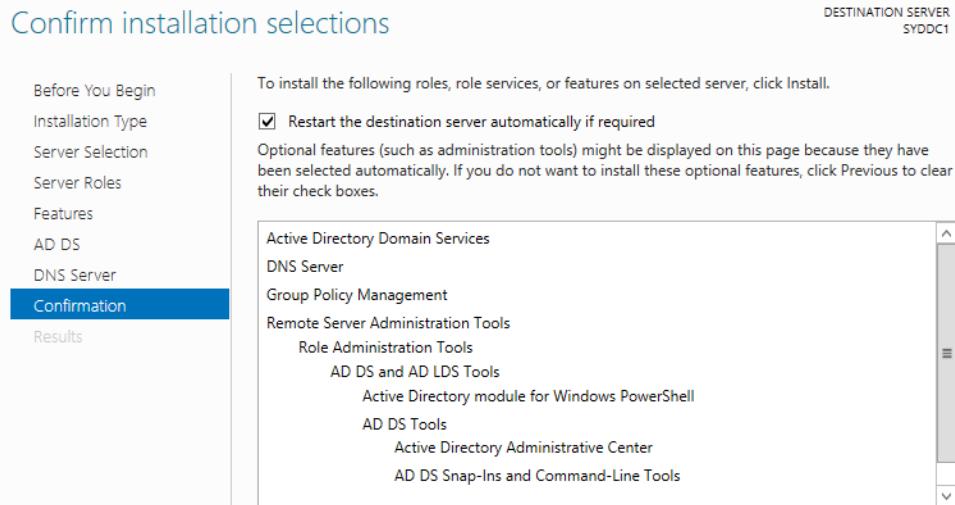


Select both **Active Directory Domain Services** and **DNS Server** from the list of roles. When prompted to add other required features and management tools as well simply click **Add Features** to accept those additions. Click **Next** to continue.



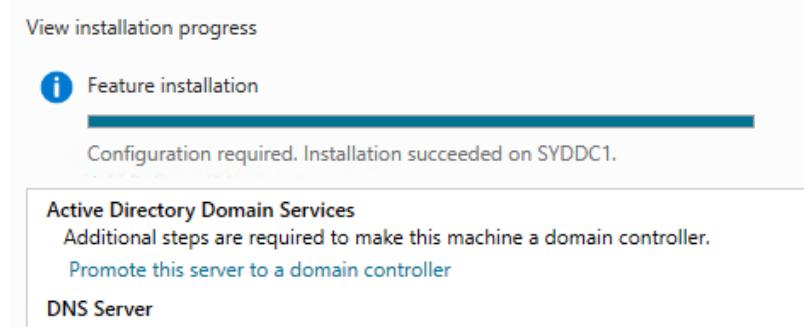
Click **Next** again to progress past the Features, AD DS, and DNS Server pages after you have read the information on each one.

On the Confirmation page tick the box to automatically restart if required, so that you don't need to manually restart it, although a restart usually will not be required at this stage anyway.



Click **Install** to begin the installation of the server roles.

When the installation has completed click the link to **Promote this server to a domain controller**.



Choose to Add a new forest. Enter your root domain name. This is where you can use your own domain name that you own as your Active Directory namespace.



**Note:** If you do not own a domain you can also use a namespace such as “domain.local”. Just be aware that using a domain name that you don’t genuinely own, may cause you some issues later on (for example we don’t recommend you use the same domain name as us, exchange2013demo.com). However if you do use a .local or similar name for now, you can still buy a domain name later if you want to establish real inbound/outbound email capabilities for your test lab environment.

Enter a Directory Services Restore Mode password and click **Next** to continue.

Domain Controller Options

TARGET SERVER  
SYDDC1

Deployment Configuration

**Domain Controller Options**

DNS Options

Additional Options

Paths

Review Options

Prerequisites Check

Installation

Results

Select functional level of the new forest and root domain

Forest functional level: Windows Server 2012 R2

Domain functional level: Windows Server 2012 R2

Specify domain controller capabilities

Domain Name System (DNS) server

Global Catalog (GC)

Read only domain controller (RODC)

Type the Directory Services Restore Mode (DSRM) password

Password:  ······

Confirm password:  ······

Disregard any DNS delegation warnings and click **Next** to continue.

DNS Options

TARGET SERVER  
SYDDC1

A delegation for this DNS server cannot be created because the authoritative parent zone cannot be found... [Show more](#) X

Deployment Configuration

**DNS Options**

Additional Options

Paths

Review Options

Prerequisites Check

Specify DNS delegation options

Create DNS delegation

Enter a NetBIOS name for the domain and click **Next** to continue.

**Additional Options**

TARGET SERVER  
SYDDC1

Deployment Configuration  
Domain Controller Options  
DNS Options  
**Additional Options**  
Paths  
Review Options  
Prerequisites Check

Verify the NetBIOS name assigned to the domain and change it if necessary

The NetBIOS domain name: EXDEMO

The default file paths are fine for a test lab environment, so click **Next** to continue.

**Paths**

TARGET SERVER  
SYDDC1

Deployment Configuration  
Domain Controller Options  
DNS Options  
**Additional Options**  
**Paths**  
Review Options  
Prerequisites Check  
Installation

Specify the location of the AD DS database, log files, and SYSVOL

Database folder: C:\Windows\NTDS ...  
Log files folder: C:\Windows\NTDS ...  
SYSVOL folder: C:\Windows\SYSVOL ...

Review your selections and click **Next** to continue.

After the prerequisites check is complete click **Install**. The server will restart to complete the process.

**Tip:** To simplify your test lab environment you might like to Group Policy Management to modify the Default Domain Policy to disable password expiry. However if you plan to open up access to your test lab from the internet (eg, Outlook Web App or ActiveSync) then complex passwords should still be used.

## Install Certificate Services

Exchange Server 2013 makes use of SSL certificates to secure client-server connectivity, such as for Outlook Web App, ActiveSync, and Outlook Anywhere.

Although SSL certificates can be purchased relatively cheaply from commercial providers, for a test lab environment you can run your own Certificate Authority (CA) and issue SSL certificates for your Exchange servers at no cost.

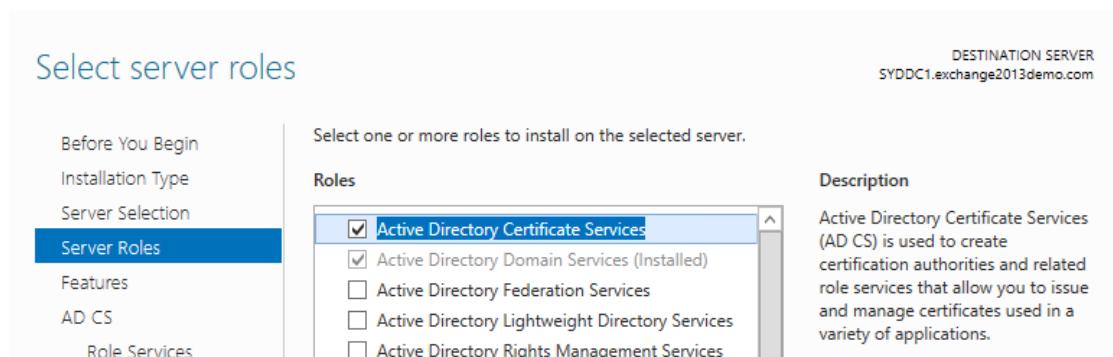
A downside to this method is that non-domain members (such as mobile devices) will not trust your CA and will display certificate trust warnings when connecting to your Exchange servers (or may not connect at all).

In Server Manager open the **Manage** menu and select **Add Roles and Features**.

Skip past the Before You Begin page and choose **Role-based or feature-based installation**. Click **Next** to continue.

Select your server from the Server Pool and click **Next** to continue.

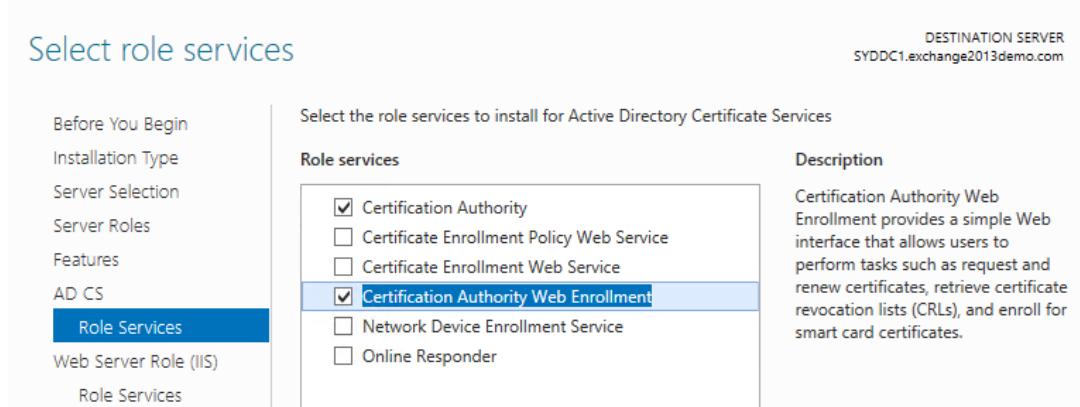
Tick the box for **Active Directory Certificate Services** and when prompted for additional required features click **Add Features**.



Click **Next** to continue past the Features and AD CS pages.

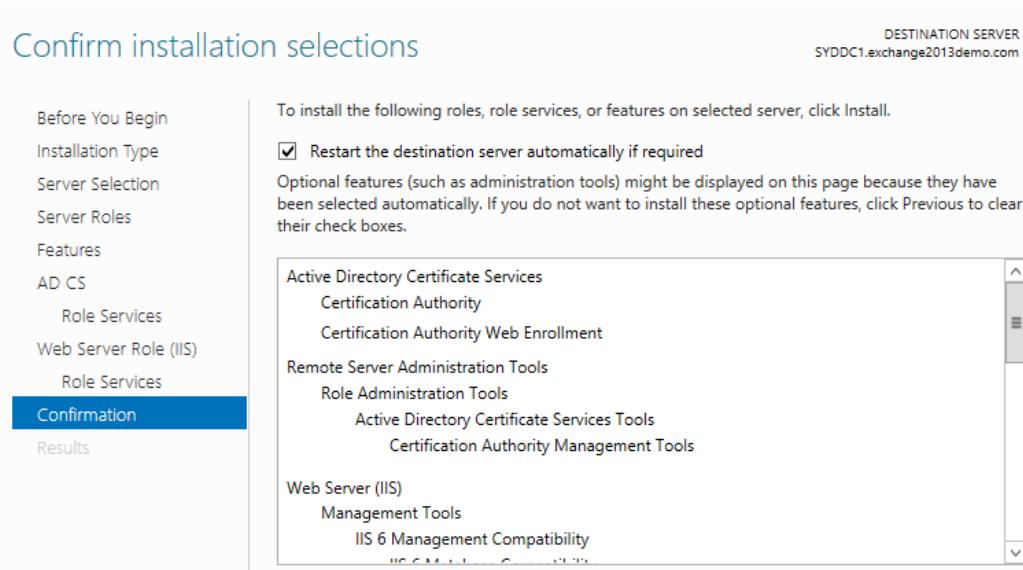
On the Role Services page tick the box for **Certification Authority Web Enrollment**, and click Add Features when prompted for additional required features.

Click **Next** to continue.

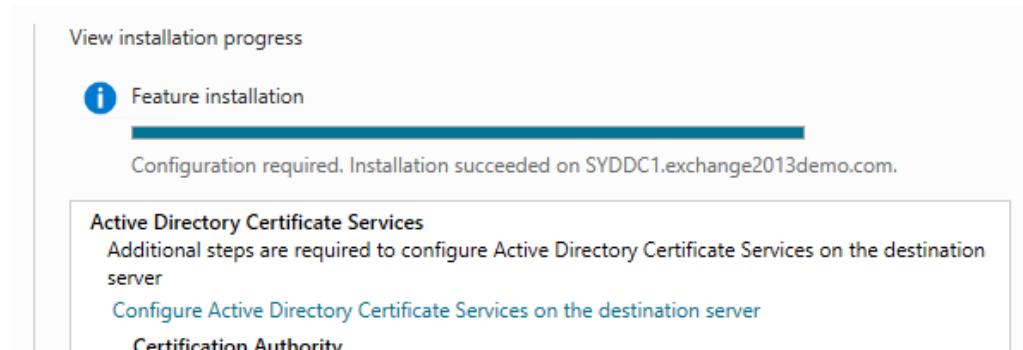


Click **Next** twice more, then on the Confirmation page tick the box to automatically restart if required (although it usually is not required).

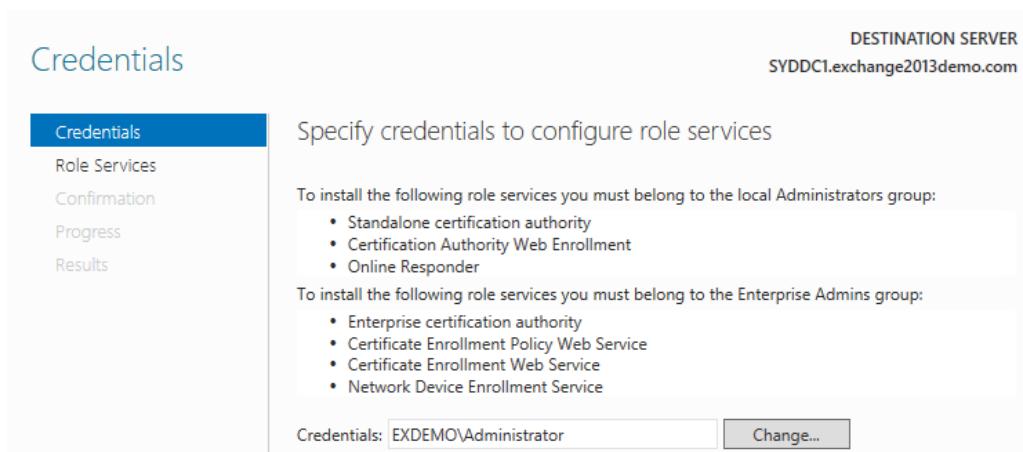
Click **Install** to begin the role installation.



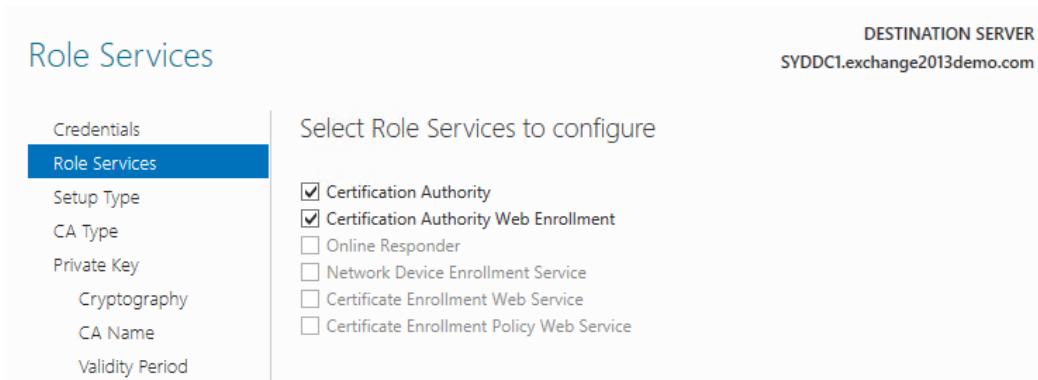
After installation has complete click the link to **Configure Active Directory Certificate Services on the destination computer**.



Use your Administrator credentials to configure the role. Click **Next** to continue.



Tick both role services and click **Next** to continue.



Continue through the wizard. Most of the selections you will be making are the default option.

- Choose to configure an Enterprise CA
- Specify a CA type of Root CA
- Create a new private key
- Accept the default cryptographic provider
- Accept the default common name for the CA (or you can change it if you wish)
- Accept the default validity period of 5 years
- Accept the default database and log locations

Confirm your selections and click **Configure** to proceed.

When the configuration has completed successfully click **Close**.

The following roles, role services, or features were configured:

**▲ Active Directory Certificate Services**

**Certification Authority**

[More about CA Configuration](#)

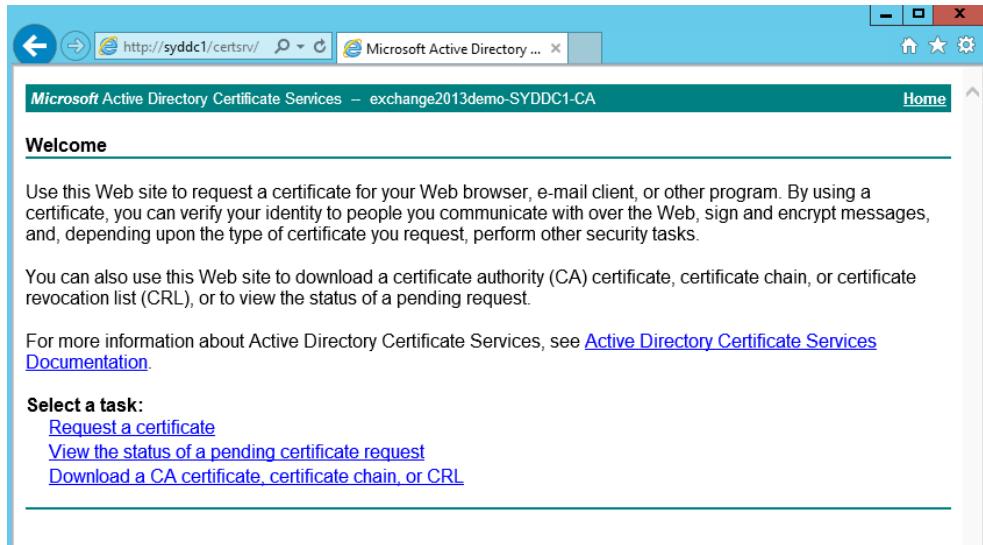
Configuration succeeded

**Certification Authority Web Enrollment**

[More about Web Enrollment Configuration](#)

Configuration succeeded

As a test you can open a web browser and go to <http://<your server name>/certsrv> to confirm that the certificate services web page loads.



The screenshot shows a web browser window titled "Microsoft Active Directory Certificate Services – exchange2013demo-SYDDC1-CA". The address bar shows the URL "http://syddc1/certsrv/". The main content area is titled "Welcome" and contains the following text:  
"Use this Web site to request a certificate for your Web browser, e-mail client, or other program. By using a certificate, you can verify your identity to people you communicate with over the Web, sign and encrypt messages, and, depending upon the type of certificate you request, perform other security tasks."  
"You can also use this Web site to download a certificate authority (CA) certificate, certificate chain, or certificate revocation list (CRL), or to view the status of a pending request."  
"For more information about Active Directory Certificate Services, see [Active Directory Certificate Services Documentation](#)".  
A section titled "Select a task:" lists three options:

- [Request a certificate](#)
- [View the status of a pending certificate request](#)
- [Download a CA certificate, certificate chain, or CRL](#)

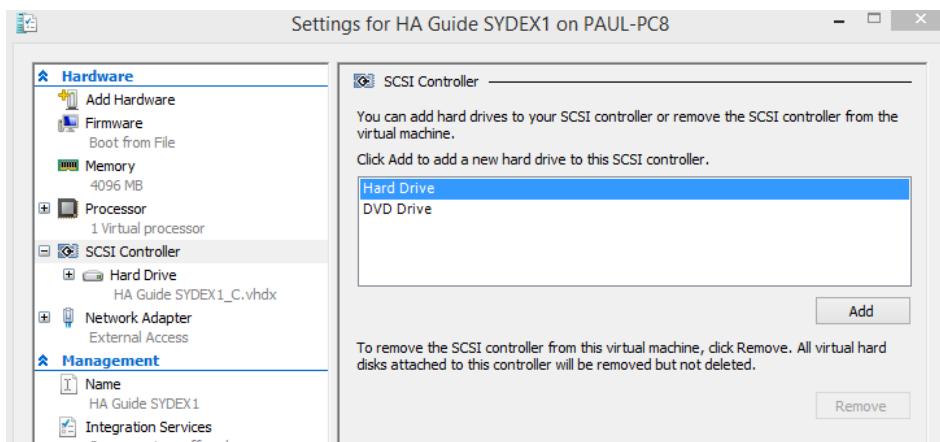
## Install Exchange Server 2013

With Active Directory set up and, optionally, a CA installed we can proceed with installation of the first Exchange Server 2013 server.

Using the same steps shown when creating the domain controller we create a new differencing disk and virtual machine, allocating at least 4Gb of memory to the virtual machine if possible, and perform the initial configuration steps for the server name, IP address, and Windows Update configuration. We can also join the server to the Active Directory domain.

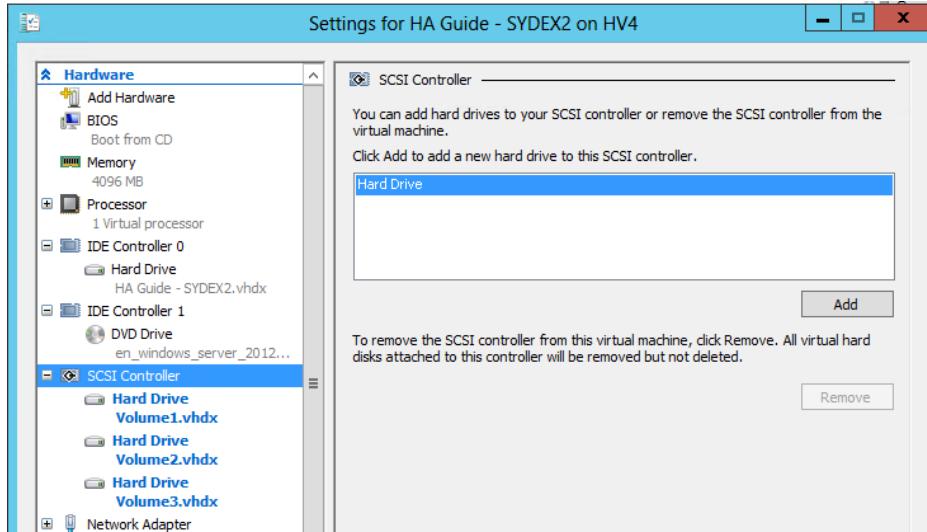
We should also add some more storage to the server to host our Exchange Server 2013 mailbox databases and transaction log files.

In the Settings of the virtual machine select the virtual SCSI Controller and add two more hard drives of equal size.

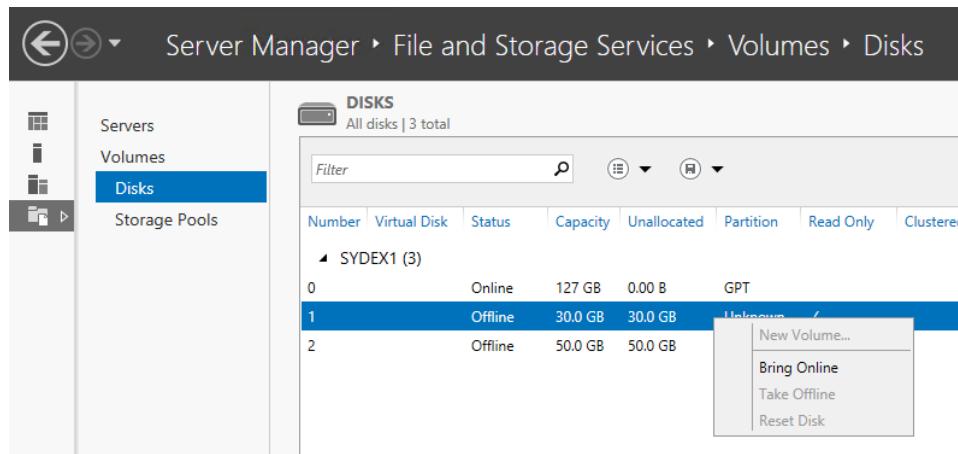


Create a new hard drive, this time as a **Dynamically Expanding** drive, which allows you to allocate a reasonable size such as 50Gb without consuming all of that space immediately on your host machine.

Repeat the process so you have three new hard drives added to the virtual machine.



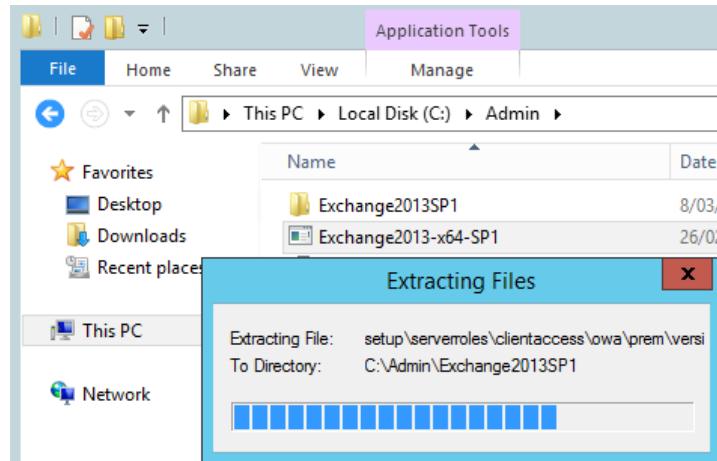
After creating the virtual hard disks open Server Manager on the server, navigate to **File and Storage Services** → **Volumes** → **Disks**, and bring the new disks online.



Then create a new volume on each disk, using the full amount of available disk space.

There is no need to assign a drive letter at this stage because we'll be covering storage layouts and configuration of mount points in the Mailbox Server HA chapter of this guide.

Create a folder on the C: drive of the server to host the installation files. Copy the Exchange Server 2013 setup file to the server and extract it to a sub-folder.



Also download the following prerequisites to the server:

- Unified Communications Managed API 4.0 Runtime<sup>31</sup>
- Microsoft Office 2010 Filter Pack 64 bit<sup>32</sup>
- Microsoft Office 2010 Filter Pack SP1 64 bit<sup>33</sup>

Open a PowerShell console and run the following command.

```
PS C:\> Install-WindowsFeature AS-HTTP-Activation, Desktop-Experience, NET-Framework-45-Features, RPC-over-HTTP-proxy, RSAT-Clustering, RSAT-Clustering-CmdInterface, Web-Mgmt-Console, WAS-Process-Model, Web-Asp-Net45, Web-Basic-Auth, Web-Client-Auth, Web-Digest-Auth, Web-Dir-Browsing, Web-Dyn-Compression, Web-Http-Errors, Web-Http-Logging, Web-Http-Redirect, Web-Http-Tracing, Web-ISAPI-Ext, Web-ISAPI-Filter, Web-Lgcy-Mgmt-Console, Web-Metabase, Web-Mgmt-Console, Web-Mgmt-Service, Web-Net-Ext45, Web-Request-Monitor, Web-Server, Web-Stat-Compression, Web-Static-Content, Web-Windows-Auth, Web-WMI, Windows-Identity-Foundation
```

Restart the server when prompted to. After restarting log back in to the server and install the prerequisites in the following order:

1. Unified Communications Managed API 4.0 Runtime
2. Microsoft Office 2010 Filter Pack 64 bit
3. Microsoft Office 2010 Filter Pack SP1 64 bit

Now we can install Exchange Server 2013 itself. Open the folder where the setup files were extracted and run Setup (setup.exe).

<sup>31</sup> <http://www.microsoft.com/en-us/download/details.aspx?id=34992>

<sup>32</sup> <http://go.microsoft.com/fwlink/p/?LinkId=191548>

<sup>33</sup> <http://go.microsoft.com/fwlink/p/?LinkId=254043>

At the updates prompt let setup connect to the internet and check for updates, just in case there are any critical security updates available.

#### MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Check for Updates?

You can have Setup download Exchange Server 2013 updates from the Internet before you install Exchange. If updates are available, they'll be downloaded and used by Setup. By downloading updates now, you'll have the latest security and product updates. If you don't want to check for updates right now, or if you don't have access to the Internet, skip this step. If you skip this step, be sure to download and install any available updates after you've completed Setup.

Select one of the following options:

- Connect to the Internet and check for updates
- Don't check for updates right now

Skip past the introduction screen, accept the license agreement, and accept the default option to use recommended settings.

Select both the Mailbox and Client Access roles, and tick the box to automatically install roles and features required for Exchange.

#### MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Server Role Selection

Select the Exchange server roles you want to install on this computer:

- Mailbox role
- Client Access role
- Management tools
- Edge Transport role
- Automatically install Windows Server roles and features that are required to install Exchange Server

Install to the default path on C: drive.

#### MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Installation Space and Location

Disk space required: 8013 MB

Disk space available: 107982.8 MB

Specify the path for the Exchange Server installation:

C:\Program Files\Microsoft\Exchange Server\V15

Choose a name for the Exchange organization. The default name of “First Organization” is fine for a test lab, or you can give it any other name you wish. Just remember that you can’t change the name later on.

MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Exchange Organization

Specify the name for this Exchange organization:

Exchange Server 2013 Demo

Apply Active Directory split permissions security model to the Exchange organization

The Active Directory split permissions security model is typically used by large organizations that completely separate the responsibility for the management of Exchange and Active Directory among different groups of people. Applying this security model removes the ability for Exchange servers and administrators to create Active Directory objects such as users, groups, and contacts. The ability to manage non-Exchange attributes on those objects is also removed.

You shouldn't apply this security model if the same person or group manages both Exchange and Active Directory. Click '?' for more information.

Leave malware scanning enabled.

MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Malware Protection Settings

Malware scanning helps protect your messaging environment by detecting messages that may contain viruses or spyware. It can be turned off, replaced, or paired with other premium services for layered protection.

Malware scanning is enabled by default. However, you can disable it if you're using another product for malware scanning. If you choose to disable malware scanning now, you can enable it at any point after you've installed Exchange.

Disable malware scanning.

Yes  
 No

Internet access is required to download the latest anti-malware engine and definition updates.

Setup will perform a readiness check for the installation.

MICROSOFT EXCHANGE SERVER 2013 SERVICE PACK 1 SETUP

? X

## Readiness Checks

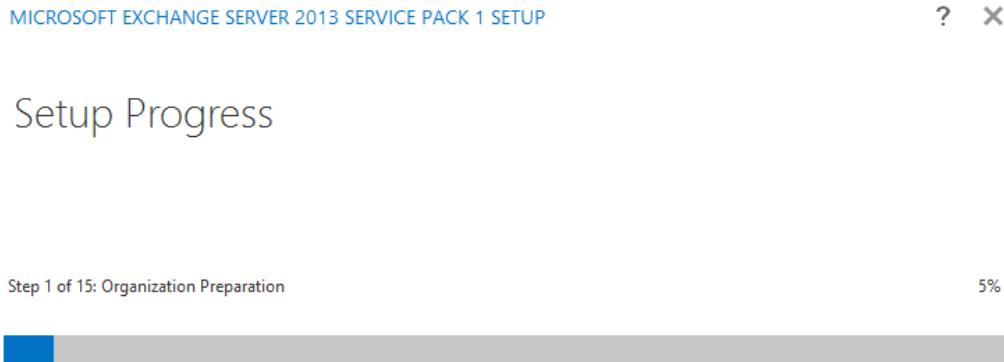
The computer will be checked to verify that setup can continue.

Configuring Prerequisites

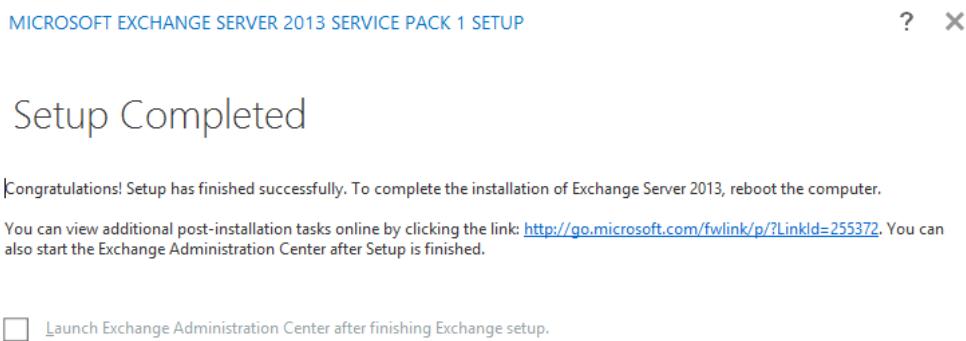
0%

You will see a warning that no Exchange 2007 or Exchange 2010 servers can be installed into this organization if you proceed with setup. This is fine for an Exchange 2013 test lab, so when you are ready to proceed click **Install** to begin the installation.

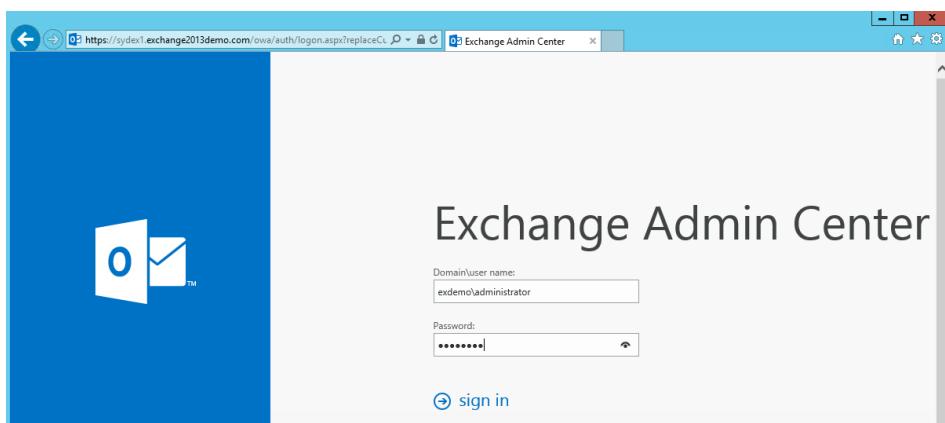
The installation itself goes through many stages and can take a long time depending on the performance of the server.



When setup is complete a restart of the server is required.



After the server restart you can test the Exchange Admin Center by opening Internet Explorer and navigating to **https://<server full-qualified name>/ecp**, and logging in with the Administrator account.



Since high availability requires at least two servers we can follow the same process again to install a second virtual machine, with a different server name, also running Exchange Server 2013.

During setup of the second server you may receive a warning about the lack of a Send Connector for your organization. This can be ignored for now.

Prerequisite Analysis 100%

Warning:  
Setup can't detect a Send connector with an address space of '\*'. Mail flow to the Internet may not work properly.  
For more information, visit: [http://technet.microsoft.com/library\(EXCHG.150\)/ms.exch.setupreadiness.NoConnectorToStar.aspx](http://technet.microsoft.com/library(EXCHG.150)/ms.exch.setupreadiness.NoConnectorToStar.aspx)

When the second server has been installed we can see both of them in the Exchange Admin Center in the Servers section.

The screenshot shows the Exchange Admin Center interface. The top navigation bar includes 'Enterprise' and 'Office 365'. Below it, the main title is 'Exchange admin center'. On the left, a sidebar lists various administrative categories: recipients, permissions, compliance management, organization, protection, mail flow, mobile, public folders, unified messaging, and servers. The 'servers' category is currently selected, indicated by a grey background. The main content area displays a table titled 'servers' with columns for NAME, SERVER ROLES, and VERSION. Two servers are listed: SYDEX1 and SYDEX2, both running Mailbox and Client Access roles, and both at Version 15.0 (Build 847.32).

NAME	SERVER ROLES	VERSION
SYDEX1	Mailbox, Client Access	Version 15.0 (Build 847.32)
SYDEX2	Mailbox, Client Access	Version 15.0 (Build 847.32)

Now we have a test lab environment set up to learn about Exchange Server 2013 high availability.

You can also configure Client Access namespaces and SSL certificates at this stage, however you may prefer to wait until you have read the Client Access server chapter first. For more information on these tasks refer to the following articles:

- [Avoiding Server Names in SSL Certificates for Exchange Server 2013<sup>34</sup>](#)
- [Exchange Server 2013 SSL Certificates<sup>35</sup>](#)

<sup>34</sup> <http://exchangeserverpro.com/avoiding-exchange-2013-server-names-ssl-certificates/>

<sup>35</sup> <http://exchangeserverpro.com/exchange-server-2013-ssl-certificates/>

# Adding Some Realism to the Test Lab

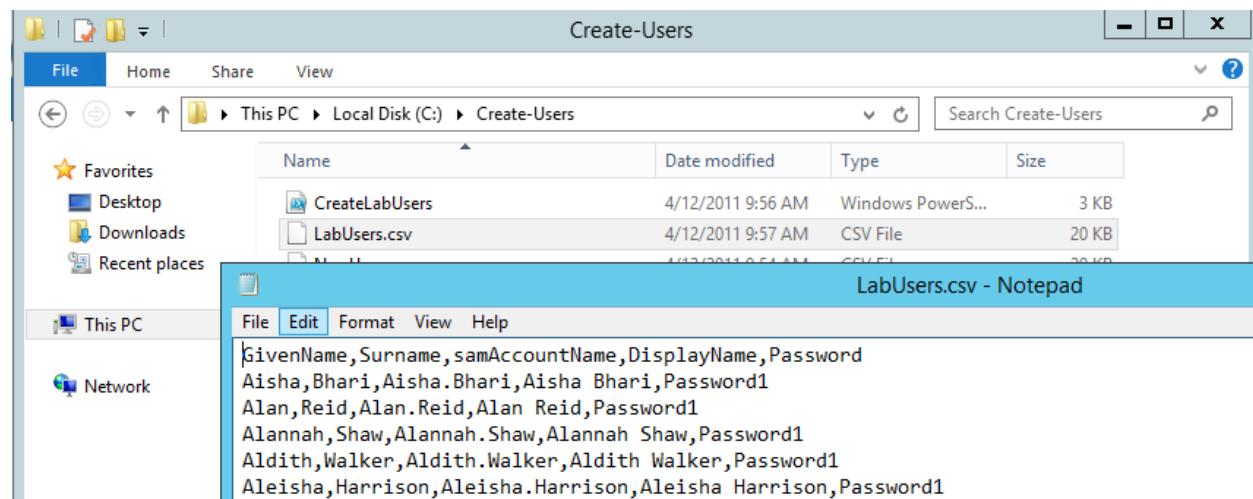
A test lab environment that sits completely idle is not very exciting to work with. Here are a few suggestions for further steps you can take to make your test lab environment more interesting, and also protect all your hard work setting the environment up.

## Creating Test Users and Mailboxes

While some people are happy to create user accounts and mailboxes manually, it is much easier to let a PowerShell script help you.

Exchange Server MVP Andy Grogan has published a [very useful script](#)<sup>36</sup> that will create a few hundred test user accounts in your Active Directory, which you can then create mailboxes for.

Before running the script it is recommended to change the password in the CSV file by running a find/replace, as the default password of “Password1” is very weak.



The test users are created in an OU named “ExchangeUsers”, so you can create mailboxes for all of those users by running a single command in the Exchange Management Shell.

```
[PS] C:\>Get-User -OrganizationalUnit ExchangeUsers | Enable-Mailbox
```

If you do this immediately after installing your test lab Exchange servers the mailboxes will be evenly distributed across the default mailbox databases created by Exchange setup.

<sup>36</sup> <http://www.telnetport25.com/2011/04/creating-lab-users-with-a-powershell-script/>

Later when you are learning about creating database availability groups and setting up database replication you can move the mailboxes to new databases, or move the databases themselves to make use of the database and log drives you provisioned for the servers.

## Simulating Email Traffic

A lot of test mailboxes is one thing but if they are not sending and receiving email then there is no point having them.

Fortunately you can simulate email traffic within your test lab, with random send/receive behaviour between mailboxes and even distribution groups if you create a few of those as well, thanks to a simple PowerShell script.

You can find the Start-MailGen.ps1 script, along with instructions for running it, on the [Exchange Server Pro website](#)<sup>37</sup>.

All of that email traffic generated by the script will create transaction logs on your servers, so you should either run it sparingly, configure circular logging for the databases, or run regular backups.

## Configuring Backups

Backing up your test lab environment is a good idea for several reasons.

- It keeps the transaction logs for the mailbox databases truncated so they do not fill up all the available disk space
- It allows you to test out backup and recovery scenarios
- It saves time if your servers become unusable and need to be restored from backup, which can be faster than rebuilding from scratch

So if you have some disk space available on your computer to store backups we recommend you set up Windows Server Backup and regular run a backup of your servers (including your domain controller).

If you are using an SSD drive for your virtual machines you may consider using a cheaper, slower SATA drive, or even an external USB drive, to store the virtual hard drives that are used as the backup destination of Windows Server Backup.

For instructions on how to set up Windows Server Backup follow the tutorial on the [Exchange Server Pro website](#)<sup>38</sup>.

## Installing a Client Machine

When it comes to testing Exchange Server 2013 from the client perspective there are a few approaches you can take:

---

<sup>37</sup> <http://exchangeserverpro.com/test-lab-email-traffic-generator-powershell-script/>

<sup>38</sup> <http://exchangeserverpro.com/backup-exchange-server-2013-databases-using-windows-server-backup/>

- Allowing regular user accounts to login to one of the servers, such as the domain controller, and running Outlook from there (not ideal, but it's only a test lab)
- Using only Outlook Web App (also not ideal, as OWA and Outlook connect over different protocols, and we'd like to test out both of them)
- Installing a virtual machine to act as the client workstation (ideal, as long as you have the resources on your host machine to run another VM)

You can install a client machine using either Windows 7 with Service Pack 1 or Windows 8.1. We would generally recommend Windows 7 as it is a little easier to interact with inside an RDP window using keyboard and mouse. However, if you don't have access to a Windows 7 ISO file to use for installation, then using the Windows 8.1 evaluation may be your only choice.

For Outlook itself we recommend installing Outlook 2013 on your client machine.

As a final tip, we recommend using a dynamically expanding virtual hard disk to save on storage space on your host machine, and using dynamic memory for the VM if necessary to allow your entire test lab environment to be running at the same time.

## Implementing Unified Messaging

As an optional step you can also configure Unified Messaging in your test lab environment. Although this tends to be more difficult to fully simulate it may still be useful if you know for sure you will be using Unified Messaging in your production environment.

If you're inside the United States, you don't need to add additional components to your existing Exchange 2013 environment, as the core components for Unified Messaging are already installed.

However, for non-US environments, you'll need to install the correct Unified Messaging Language pack to ensure the voice prompts and recognition are customized for your region. You can download the UM Language Packs from [the Microsoft website](#)<sup>39</sup>.

After downloading the UM language packs, install them on each Exchange 2013 server hosting the Mailbox role. You can perform this by copying the downloaded language pack to the server and running the self-extractor (e.g. UMLanguagePack.en-GB.exe) or by using the **setup.exe /AddUmLanguagePack** command.

## Certificate Configuration

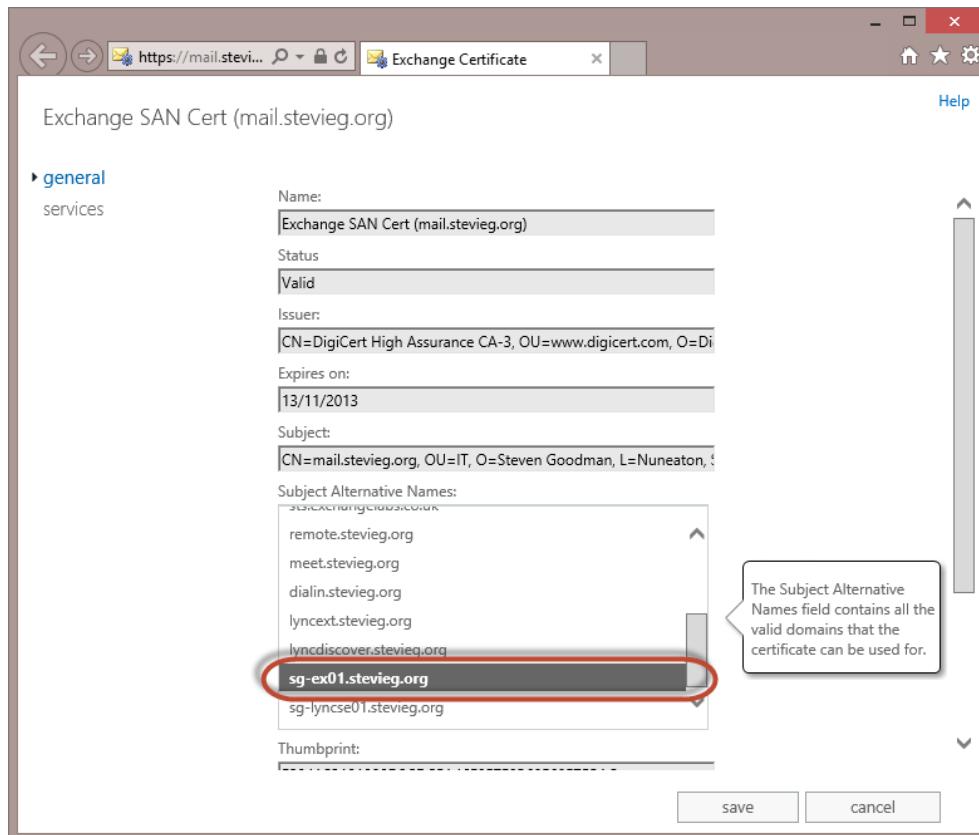
In the example below, we'll verify that the Exchange Server name is on the existing certificate by first accessing the Exchange Admin Center, then navigating to **Servers → Certificates** and selecting our existing SAN certificate from the list.

---

<sup>39</sup> <http://www.microsoft.com/en-us/download/details.aspx?id=35368>

The screenshot shows the Exchange admin center interface. On the left, there's a navigation pane with various categories like recipients, permissions, compliance management, organization, protection, mail flow, mobile, public folders, unified messaging, and servers. The 'servers' category is selected. The main area displays a table of certificates. One row is highlighted for the 'Exchange SAN Cert (mail.stevie...)', which is listed as 'Valid' and expires on '13/11/2013'. Other entries include 'Microsoft Exchange Server Auth ...', 'Exchange Delegation Federation', 'Microsoft Exchange', and 'WMSVC', all marked as 'Valid'. To the right of the table, there's a status summary: 'Status' (Valid), 'Expires on: 13/11/2013', a 'Renew' link, and notes about assigned services: 'IMAP, POP, IIS, SMTP'.

We'll then click **Edit** to open the Exchange Certificate dialog window, and then on the **General** tab verify the list of Subject Alternative Names. As you'll see in the example below, our certificate includes the correct server name:



After verification or update of certificates on each Exchange Server, we're ready to change each Exchange Server's UM settings from the default non secure startup mode of TCP to a minimum of Dual, which allows TLS and normal TCP SIP communications.

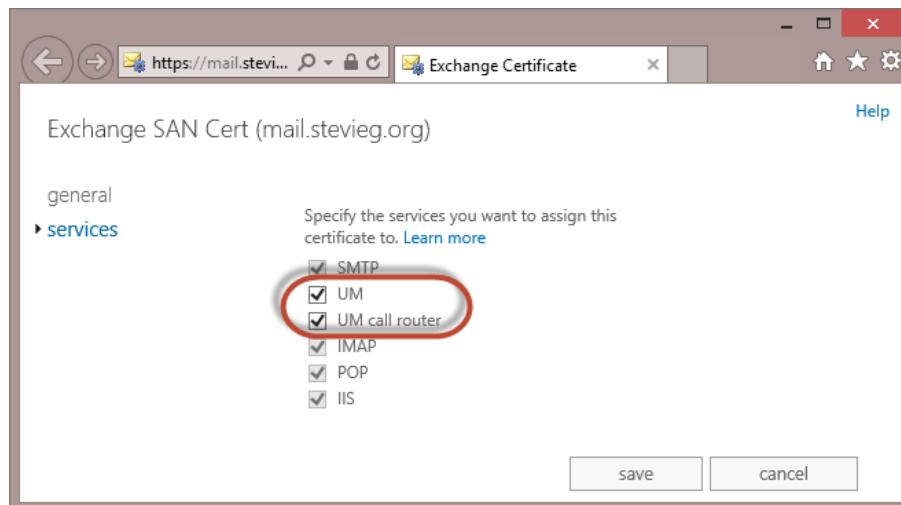
From the Exchange Management shell, execute the following commands, substituting the relevant server name, against each server.

```
[PS] C:\>Set-UMService SYDEX1 -UMStartupMode Dual  
[PS] C:\>Set-UMCallRouterSettings SYDEX1 -UMStartupMode Dual
```

After executing the commands, you'll be notified that the changes will take effect only after restarting the relevant Unified Messaging or Unified Messaging Call Router service. That's fine, because we need to perform a couple of more tasks before restarting services.

We can do that against each server by revisiting the relevant Exchange Certificate dialogue in the EAC, and navigating to the **Services** tab.

For multi-role servers select both **UM** and **UM call router**. For servers only hosting the Mailbox server role, select **UM** and for servers only hosting the Client Access role, select **UM call router**, then choose **Save**.



## Creating the Unified Messaging Dial Plan

Our next step is to create a Dial Plan. We'll create our UM dial plan using the EAC, by navigating to **Unified Messaging** and selecting the **UM Dial Plans** tab.

Select the **Add** button as shown below to initiate creation of the dial plan.

The screenshot shows the Exchange admin center interface. On the left, there's a navigation pane with links like recipients, permissions, compliance management, organization, protection, mail flow, mobile, public folders, unified messaging (which is selected and highlighted in blue), and servers. The main area is titled "UM dial plans" and shows a table with columns: NAME, EXTENSION LENGTH, and URI TYPE. A note at the top says: "Unified Messaging dial plans define the format for telephone numbers in your organization. For UM to answer calls for your users, you have to set up at least one dial plan. [Learn more](#)". Below the table, it says "There are no items to show in this view." At the bottom right of the table area, it says "0 selected of 0 total".

We'll then enter appropriate details to match our Lync infrastructure.

The dialog box is titled "new UM dial plan". It contains the following fields:

- \*Name: Lync
- \*Extension length (digits): 4
- \*Dial plan type: SIP URI
- \*VoIP security mode: SIP secured
- \*Audio language: English (United Kingdom)
- \*Country/Region code: 44

A note at the bottom says: "After you click Save, select this dial plan and click Edit to configure dial codes, Outlook Voice Access, voice mail settings, and dialing rules."

At the bottom right are "save" and "cancel" buttons.

The **Name** value can be anything you choose, so in our case we'll keep it simple and call it *Lync*.

The **Extension length** corresponds to the number of digits you use internally for Lync - therefore we'll set ours to 4 to match our internal Lync extensions.

The **Dial plan type** should be set to *SIP URI*.

The **VoIP security mode** should be set to *SIP secured*.

Finally, choose appropriate **Audio language** and **Country/region codes**, then press **Save**.

With our dial plan created, we'll now need to edit the basic configuration to specify at a minimum the dial-in number users can dial to access their voicemail.

Choose the new Dial Plan in the EAC, then select **Edit**.

UM dial plans   UM IP gateways

Unified Messaging dial plans define the format for telephone numbers in your organization. For UM to answer calls for your users, you have to set up at least one dial plan. [Learn more](#)

NAME	EXTENSION LENGTH	URI TYPE
Lync	4	SIP URI

In the Dial Plan dialog window that opens, select **Configure** to open the basic dial plan settings.

Lync Help

Configure settings for this dial plan, including UM mailbox policies, auto attendants, and hunt groups.

**UM Dial Plan**

Name:	Lync
Dial plan type:	SIP URI
Extension length (digits):	4

To configure dial codes, Outlook Voice Access, voice mail settings, and dialing rules for this dial plan, click **Configure**.

**configure**

**UM Mailbox Policies**

+   ✎   ━   ⌂

Next, navigate to the **Outlook Voice Access** tab and scroll down to **Outlook Voice Access Numbers**. We'll enter the number that users will dial to access voicemail. This may be (as in our case - 6002) an internal only number within the dial plan, or alternatively you may also specify the external dial in number users can use from any PSTN line.

After entering the number and adding it to the list, choose *Save*:



With our new Dial Plan created and basic settings configured, we'll then choose to associate the Dial Plan with each Exchange Server within our site, running the following commands against each Exchange Server specifying the name of the dial plan (in our case, "Lync") and the server name.

For multi-role servers, we'll use both commands, but Client Access servers will only require the Set-UMCallRouterSettings cmdlet and Mailbox servers will only require the Set-UMService cmdlet.

```
[PS] C:\>Set-UMCallRouterSettings SYDEX1 -DialPlans "Lync"  
[PS] C:\>Set-UMService SYDEX1 -DialPlans "Lync"
```

Finally to apply the new certificate settings and Dial Plan bindings to the UM service, we'll use the following command to restart the UM services on each Exchange server.

```
[PS] C:\> Get-Service MSExchangeUM* | Restart-Service
```

## Configuring Exchange for Lync Integration

As you'd hope, Microsoft make it easier to integrate Lync with Exchange UM than other IP-PBX providers by providing a script that helps make the job easier. The ExchUCUtil.ps1 script performs a couple of major functions:

- Configures permissions to grant access to UM-related Exchange components by the Lync server.
- Creates Unified Messaging IP gateways to define in Exchange which Lync servers to communicate with.

To execute this script, we'll open the Exchange Management Console, and first navigate to the Exchange Scripts directory.

```
[PS] C:\> cd $exscripts
```

Then we'll execute the ExchUCUtil.ps1 script.

```
[PS] C:\Program Files\Microsoft\Exchange Server\V15\Scripts> .\ExchUCUtil.ps1
```

The script produces quite lengthy output.

```
Machine: SG-EX01.stevieg.org
[PS] C:\Program Files\Microsoft\Exchange Server\V15\Scripts> .\ExchUCUtil.ps1
Using Global Catalog: GC://DC=stevieg,DC=org

Configuring permissions for stevieg.org\RTCUniversalServerAdmins ...
StevieG: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
UM DialPlan Container: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
UM AutoAttendant Container: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
Administrative Groups: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.

Configuring permissions for stevieg.org\RTCComponentUniversalServices ...
StevieG: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
UM DialPlan Container: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
UM AutoAttendant Container: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.
Administrative Groups: The appropriate permissions have been granted for the Lync servers and Administrators to be able to read the UM dial plan and auto attendants container objects in Active Directory. No new permissions have been added to the container objects.

Configuring UM IP Gateway objects...
Pool: sg-lyncse01.stevieg.org
A UMIPGateway already exists in Active Directory for the Lync Server pool. A new UM IP gateway wasn't created for the pool.
IsBridgeEnabled: False
MessageWaitingIndicatorAllowed: True
OutcallsAllowed: True
WARNING: The command completed successfully but no settings of 'sg-lyncse01' have been modified.
Dial plans: Lync

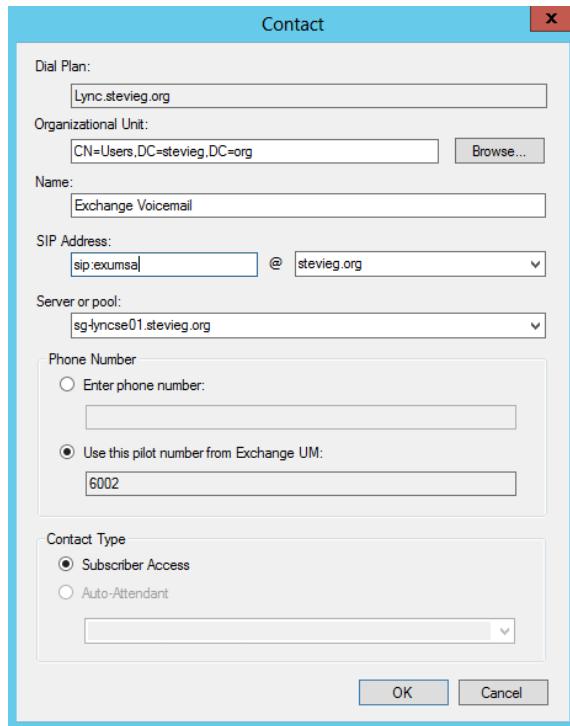
Permissions for group stevieg.org\RTCUniversalServerAdmins
Object Name Access Rights Configured
StevieG ListChildren True
UM DialPlan Container ListChildren, ReadProperty True
UM AutoAttendant Container ListChildren, ReadProperty True
Administrative Groups ListChildren, ReadProperty True

Permissions for group stevieg.org\RTCComponentUniversalServices
Object Name Access Rights Configured
StevieG ListChildren True
UM DialPlan Container ListChildren, ReadProperty True
UM AutoAttendant Container ListChildren, ReadProperty True
Administrative Groups ListChildren, ReadProperty True

PoolFqdn UMIPGateway DialPlans
sg-lyncse01.stevieg.org sg-lyncse01 <Lync>
```

In particular, pay attention to the permissions granted and verify that the PoolFQDN, UMIPGateway and associated DialPlans are correct.

On the Contact dialog window that opens, enter relevant settings appropriate for the new contact:



In the example above, we've chosen the Organizational Unit where the contact object will be stored in AD, entered a friendly name of Exchange Voicemail and set the SIP Address to sip:exumsa to represent that it's Exchange Unified Messaging Subscriber Access. If you've configured a similar contact (or plan to) for Exchange Online, you should ensure that these can be distinguished, at least by an administrator.

Other settings such as the Server/Pool and Phone Number are picked up automatically. Verify these are correct, then select OK to save the new contact. You should see your new contact listed as type "SA" in the Exchange UM Integration Utility after successful creation.

## Enable Mailboxes for Unified Messaging

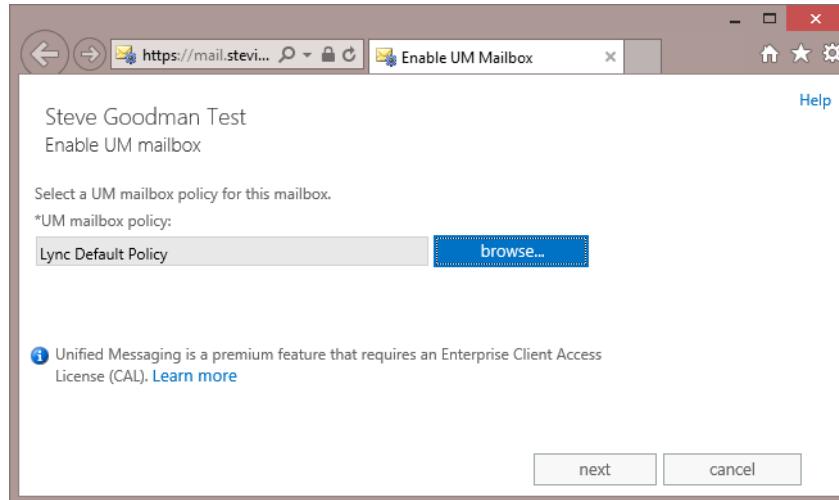
With Lync and Exchange integration configured, we're now ready to enable Mailboxes for Unified Messaging functionality.

To perform this task, we'll hop back over to the Exchange Admin Center and navigate to **Recipients → Mailboxes** and select an appropriate mailbox.

After selecting a Mailbox, scroll down to **Phone and Voice Features** and under the Unified Messaging heading, choose **Enable**, as shown below.

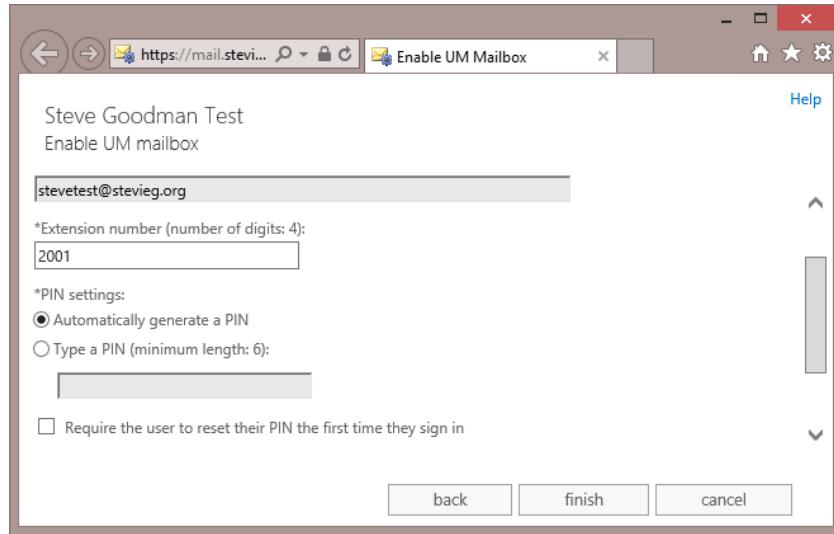
The screenshot shows the Exchange admin center interface. On the left, there's a sidebar with categories like recipients, permissions, compliance management, organization, protection, mail flow, mobile, public folders, unified messaging, and servers. The 'recipients' category is currently selected. On the right, a main pane displays a list of mailboxes. One mailbox, 'Steve Goodman Test', is selected and highlighted with a dark grey background. To the right of the list, there are several options: 'Phone and Voice Features' (with 'Unified Messaging: Disabled' and an 'Enable' link circled in red), 'Mobile Devices', 'Disable Exchange ActiveSync', 'View details', and 'In-Place Archive'. At the bottom of the list, it says '1 selected of 8 total'.

After choosing to enable UM, the Enable UM Mailbox dialog window will open. We'll select the automatically created Lync Default Policy UM Mailbox Policy that corresponds to our Dial Plan, and then choose **Next**.



Finally, we'll enter the SIP address, the extension (if it's not already set in Active Directory in the IPPhone attribute) and either choose a new PIN, or automatically generate a user PIN.

After choosing these options, choose **Finish**.

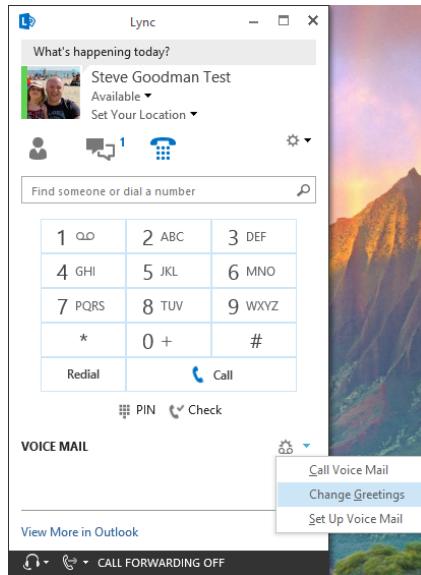


After choosing Finish, the mailbox will be enabled for UM functionality and the user will receive an email containing their PIN and other relevant details, such as the Subscriber Access number.

## Testing Unified Messaging in Lync and Outlook Web App

After signing out of Outlook Web App and Lync and signing back in, we'll now see that both present us with options for accessing and managing Voicemail.

In Lync, we'll find these settings on the dial pad, in the Voice Mail section. From here we can set up Voice Mail options, dial into Outlook Voice Access or choose Set up Voice Mail to jump to the relevant options page in Outlook Web App.



If we then choose the Set Up Voice Mail option or navigate to Options>Phone>Voice Mail in Outlook Web App, we'll also see new options for managing call answering rules, PIN settings, voice mail preview, play on phone, greetings and notifications.

The screenshot shows the Microsoft Outlook Web App interface. The top navigation bar includes the 'Outlook Web App' logo, user name 'Steve Goodman Test', and a help icon. The left sidebar has a 'options' menu with links for account, organize email, groups, site mailboxes, settings, phone (which is selected), block or allow, and apps. The main content area has tabs for 'voice mail', 'mobile devices', and 'text messaging'. Under 'voice mail', there's a 'reset PIN' section with instructions and a link to 'Reset my voice mail PIN...'. To the right is a 'greetings' section with two radio button options: 'Default voice mail greeting' (selected) and 'Greeting that lets callers know you'll be away for an extended period of time'. Below that is an 'Outlook Voice Access' section with a text input field 'Dial this phone number:' containing 'stevetest@stevieg.org' and a 'dial' button. At the bottom is a blue 'save' button.

## Lab Guide Summary

Having a fully functional test lab environment will be an enormous help to you as you work through this guide and other Exchange Server 2013 training in future.

Create the test lab that you're able to fit within your resources and budget. With so many different options available now for hosting a lab, such as cheap server hardware, powerful laptops and desktops, or even cloud services like Microsoft Azure, a test lab environment is within reach for most people.

However if you still are not able to create a test lab environment you can still learn a lot just from reading this guide and applying it to your day to day job when appropriate.

*Thank you for reading.*

*Paul, Michael and Steve*