Professional Expertise Distilled

Microsoft DirectAccess Best Practices and Troubleshooting

Secure and efficient functioning of your DirectAccess environment

Foreword by Richard Hicks, MVP

Jordan Krause

[PACKT] enterprise
PUBLISHING
professional expertise distilled

Microsoft DirectAccess Best Practices and Troubleshooting

Secure and efficient functioning of your DirectAccess environment

Jordan Krause

[PACKT] PUBLISHING enterprise
professional expertise distilled

BIRMINGHAM - MUMBAI

Microsoft DirectAccess Best Practices and Troubleshooting

First published: October 2013

Production Reference: 1071013

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78217-106-5

`www.packtpub.com`

Cover Image by Fereze Babu (`mail@feroze.me`)

Credits

Author
Jordan Krause

Reviewers
Shannon Fritz
Richard Hicks

Acquisition Editor
Vinay Argekar

Commissioning Editor
Neha Nagwekar

Technical Editors
Novina Kewalramani
Rohit Kumar Singh

Project Coordinator
Sherin Padayatty

Proofreader
Clyde Jenkins

Indexer
Mariammal Chettiyar

Graphics
Yuvraj Mannari

Production Coordinator
Aparna Bhagat

Cover Work
Aparna Bhagat

Foreword

Microsoft DirectAccess is a revolutionary remote access solution for managed (domain-joined) Windows clients. DirectAccess provides always-on corporate network connectivity, enabling remote users to securely access on-premises data and applications anywhere they have a connection to the public Internet. Many mistakenly believe that DirectAccess is itself a protocol. It is not. DirectAccess leverages multiple Microsoft technologies to deliver this service, such as Active Directory, IPsec, IPv6, digital certificates, and more. Harnessing the power of Windows Server 2012 and Windows 8 Enterprise edition, DirectAccess represents a paradigm shift in the way we think about providing remote access. Traditional Virtual Private Networking (VPN) solutions require the user to proactively initiate a connection back to the corporate network when they need to access corporate resources. By contrast, DirectAccess is seamless and transparent, and does not require any input from the user to establish remote network connectivity. Through the use of Connection Security Rules in the Windows Firewall with Advanced Security (WFAS), IPsec tunnels are established automatically in the background any time the user has an active Internet connection. A distinct advantage that DirectAccess has over VPN is that DirectAccess is bidirectional, allowing hosts on the corporate intranet to initiate connections outbound to connected DirectAccess clients. This allows system administrators to "manage out" and enables help desk administrators to initiate remote desktop sessions or security administrators to conduct vulnerability scans, among other things. DirectAccess fundamentally extends the corporate network to the remote user, wherever they may be located.

DirectAccess has been around for a few years, originally appearing as a feature of the Windows Server 2008 R2 operating system. Windows Server 2008 R2 DirectAccess wasn't widely deployed, as it carried with it very steep infrastructure requirements in order to support DirectAccess, including the requirement for a Public Key Infrastructure (PKI) for management of digital certificates and IPv6 for network layer transport. My first experience with DirectAccess came when Forefront Unified Access Gateway (UAG) 2010 was released. UAG included support for the DirectAccess role, and also included new features that eliminated the need to deploy IPv6 internally to take advantage of the solution.

As a Microsoft Most Valuable Professional (MVP) in the Forefront discipline, I began to deploy Forefront UAG for DirectAccess on a regular basis. With the release of Windows Server 2012, DirectAccess is now fully integrated into the operating system, and the adoption rate is accelerating faster. Today, I spend most of my time deploying Windows Server 2012 DirectAccess solutions for some of the largest organizations in the world.

I met Jordan Krause a few years ago when he was first awarded the MVP from Microsoft. Our MVP group is small and tight-knit, and from the beginning Jordan fit right in. He had a wealth of knowledge and experience with DirectAccess and freely shared this with the rest of us in the group. All of us in the DirectAccess community have gained important knowledge from Jordan. With this book, Jordan is now able to share his valuable experience with the rest of the world. This book is focused on sharing real-world, practical advice for deploying DirectAccess in the best possible way for your given deployment model. Jordan pulls no punches, and isn't afraid to tell you when you shouldn't do something, even if it is possible! He provides valuable context to help you with your implementation, and makes sure that you avoid the common pitfalls and mistakes that many engineers who are new to DirectAccess invariably make. If you're going to deploy Windows Sever 2012 DirectAccess now or in the future, you'll definitely want to read this book first.

Enjoy!

Richard Hicks

Director of Sales Engineering at Iron Networks, Inc.

About the Author

Jordan Krause is a Microsoft MVP in Enterprise Security, and specializes in DirectAccess, which is a part of Forefront Unified Access Gateway (UAG) 2010 and Unified Remote Access (URA) in Windows Server 2012. As a Senior Engineer and Security Specialist for IVO Networks, he spends the majority of each workday planning, designing, and implementing DirectAccess for companies all over the world.

Committed to continuous learning, Jordan holds Microsoft certifications as an MCP, MCTS, MCSA, and MCITP Enterprise Administrator. He regularly writes tech notes and articles about some of the fun and exciting ways that DirectAccess can be used, which can be found at `http://www.ivonetworks.com/news/`.

He also strives to spend time helping the DirectAccess community, mostly by way of the Microsoft TechNet forums. Jordan is always open to direct contact for answering questions or helping out in any way that he can, so don't hesitate to head over to the forums and find him personally.

Huge thanks to my family for taking more on their plates while I worked on this. Laura, Grace, and Jackson—you are my motivation for doing what I do! Another big thank you to my family at IVO; without the opportunities you have provided, I may never have heard the word DirectAccess.

About the Reviewers

Shannon Fritz is an Infrastructure Architect and regional leader in Remote Connectivity solutions, including DirectAccess, Remote Desktop Services, and supporting technologies such as Hyper-V and Active Directory. Shannon is the Datacenter and Azure Team Lead for Concurrency's Infrastructure Practice, a systems integrator who is solely focused on Microsoft solutions.

Richard Hicks (MCP, MCSE, MCTS, and MCITP Enterprise Administrator) is a network and information security expert specializing in Microsoft technologies. As a four-time Microsoft Most Valuable Professional (MVP), he has traveled around the world speaking to network engineers, security administrators, and IT professionals about Microsoft edge security and remote access solutions. Richard has nearly two decades of experience working in large scale corporate computing environments, and has designed and deployed perimeter defense and secure remote access solutions for some of the largest companies in the world. He blogs extensively about Microsoft edge security and remote access solutions, and is a contributing author at popular sites such as `WindowSecurity.com`, `ISAserver.org`, and the Petri IT Knowledgebase. In addition, he is a Pluralsight author and has served as the technical reviewer on several Windows server and network security books. Richard is the Director of Sales Engineering for Iron Networks, a Microsoft OEM partner developing secure remote access, network virtualization, and converged cloud infrastructure solutions. He's an avid fan of Major League Baseball and in particular the Los Angeles Angels (of Anaheim!), and also enjoys craft beer and single malt Scotch whisky. Born and raised in beautiful, sunny Southern California, he still resides there with Anne, the love of his life and wife of 27 years, along with their four children. You can keep up with Richard by visiting `http://www.richardhicks.com/`.

Table of Contents

Preface

If you have walked someone through installing and configuring a VPN connection over the phone, you might be a VPN Administrator.

If you have tried explaining to someone that they have to come into the office before they can log into their laptop, because you reset their password but they can't use it... you might be a VPN Administrator.

If you are aware that home subnets might have the same IP ranges as your corporate subnets, and the reason why that is bad...you might be a VPN Administrator.

If you cringe when a laptop is plugged into the network after being gone on vacation for a couple of weeks…well, you might be a network security admin yelling at your VPN Administrator.

If you want to rid yourself of all these issues and give users a completely seamless connection that they don't even have to know exists...you might get a big bonus check. Oh, and you might be a DirectAccess Administrator!

DirectAccess rocks

I always said if I had an opportunity to write something about DirectAccess, I would at some point say "DirectAccess rocks", and so there it is. I spend at least part of everyday describing the technology to new folks and comparing it to a traditional VPN connection, but there really is no comparison. Your users either have to launch a VPN client, or they don't. You either have to install and configure and update that VPN client software, or you don't. You either wait around for employees to choose to connect their VPN so that you can push security updates and settings at them, or you don't. DirectAccess is basically automatic VPN, and after years of talking about it on phone calls and at shows, I am convinced that I can get anyone interested in it. Though the technology has been around in one flavor or another for four years now, it is still a brand new concept to many, and all it takes is a few minutes to get anyone who has ever used a VPN interested in never having to use one again.

So many options

Unfortunately, a lot of DirectAccess implementations are halted before they even start, and it's really unnecessary. Part of the problem is IPv6; as soon as admins hear that DirectAccess uses IPv6, they immediately discount it as something that does not apply to them. This is completely untrue; you don't actually have to know anything about IPv6 or use it at all inside your network to get DirectAccess working! Another "problem" that I address all the time is that there are so many different ways in which DirectAccess can be implemented, how is one supposed to sift through and figure out what is best for them? This is a large part of the intention of this book, to clear the air on the options that are out there, and particularly address them from a set of "Best Practices" glasses. We are going to talk about specific settings and some general ideology about how to make DA work its hardest for you and your organization, and have a little fun along the way.

Take it from me

Implementing DirectAccess is quite literally my day job, and the ideas and steps outlined in this book reflect my own experience and knowledge directly from the field. We all know that implementation of technology rarely goes according to plan, and I hope that you can take some of the speed bumps that I have overcome along the way and apply them to your own situations to make your installation as seamless as possible.

Which flavor of DirectAccess are you talking about?

If you have done some reading on DA, you may be aware that there are two different server platforms which can provide DirectAccess. Well, there are three technically, but the original iteration in native Server 2008 R2 was quite difficult to handle, and I have yet to run across a network with it running. The other two, of which I still actively install both very regularly, are **UAG DirectAccess** and **Server 2012 DirectAccess**. As you can infer from the name, the latter runs on Server 2012 and is simply a role that you can add into Windows (don't do this until you read *Chapter 1*, *DirectAccess Server Best Practices*). UAG, on the other hand, is a software platform that needs to be installed on top of Server 2008 R2. If one is Server 2008 R2 and the other is Server 2012, why would anybody still be doing UAG? Both platforms provide DirectAccess connection for Windows 7 and Windows 8 client computers, but the two platforms handle non-DirectAccess machines very differently.

In Server 2012, you have the option to provide regular RRAS VPN connectivity, so if you still have Windows XP clients or Macs or smartphones with a VPN software client installed, you can connect those guys through the server via regular VPN. This may be beneficial, or it may be downright scary, depending on your perspective. With the UAG platform, you again have Windows 7 and Windows 8 running DirectAccess, and you also have the ability to publish SSLVPN portals out on the Internet. These portals enable browser-based access from home computers, kiosks, mobile devices, and so on, in a selective, locked-down way. There are already great books available on UAG and everything that it stands for so I won't say any more than that, but I wanted to make the point that UAG is still today a valid option for implementing DirectAccess, if those other features are important to you. Or you could, of course, have a server running UAG for those down-level clients, and a separate server running DirectAccess on Server 2012, if that is your preference.

Anyway, the point of this section is to simply say that the information contained within this book applies specifically to Server 2012 DirectAccess, but all of the concepts can absolutely also apply to UAG DirectAccess. I used Server 2012 to create my command output, screenshots, and for all of the verbiage within the book. But all of the security concepts and guides to troubleshooting client-side scenarios really apply to either solution.

Let's get rolling

I had a lot of fun putting this together, and I hope you get some enjoyment out of reading it. I genuinely believe that DirectAccess is the future of remote access. It is one of those rare gems in the IT world where your department can receive a well-deserved slap on the back by the end users and executive team. Trust me, it's that cool.

What this book covers

Chapter 1, *DirectAccess Server Best Practices*, describes the step-by-step procedure you should take to prepare your DirectAccess server. Following the procedures listed here will ensure that your server adheres to critical security practices.

Chapter 2, *DirectAccess Environmental Best Practices*, brings detail to the infrastructure and environmental considerations that need to be taken when implementing DirectAccess. Many common implementation questions are also addressed.

Chapter 3, *Configuring Manage Out to DirectAccess Clients*, brings some clarity to that mysterious thing they call ISATAP. Most of us have heard of it, and maybe know that it has something to do with managing your DirectAccess clients, now let's take an in-depth look into whether or not you actually need it, and how to correctly utilize it when you do.

Chapter 4, General DirectAccess Troubleshooting, will enable you to make sense of those client log files, pointing out the important sections and what they mean. With the information provided here, you should be able to diagnose a connection within a matter of seconds.

Chapter 5, Unique DirectAccess Troubleshooting Scenarios, is an interesting walk through some of the cases I have worked which you may not encounter every day. Understanding the causes and resolutions to these issues could be the difference between minutes and days when it comes to diagnosing these issues.

What you need for this book

Many of you will already have DirectAccess in your environment, and as such you probably already have everything you need. I suppose that is not necessarily true, as after reading through some of the environmental considerations, you may choose to enforce some additional measures that could mean you introduce a couple of new items in your network, but I will let the chapters themselves speak to that. For anyone new to this technology, DirectAccess is heavily integrated with the domain, utilizing groups and Group Policies for configuration, so running a network where Active Directory exists is a must. You will also need a server which you are planning to turn into your DirectAccess server, running Windows Server 2012. Any client computers that you want to connect through this server must be Windows 7 Enterprise, Windows 7 Ultimate, or Windows 8 Enterprise, and it would be a good idea to have at least one of those guys ready so that you can test when finished with the configuration.

Who this book is for

This book will be of interest to any existing DirectAccess administrator, and to anyone interested in learning more about the technology before diving in for themselves. Although the topics covered here are geared for the specific purposes of enhancing DirectAccess, I also encourage any administrator who has the unfortunate task of dealing with a tradition VPN on a day-to-day basis absolutely do all the reading they can on this technology, and cut over to it as quickly as possible to save themselves time, money, and headaches.

Conventions

In this book, you will find a number of styles of text that distinguish among different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Now, before you get all huffy with me, yes I do know about the new feature in Server 2012 DirectAccess that allows the second encryption to be `null`".

Any command-line input or output is written as follows:

```
Route add -p 192.168.2.0 mask 255.255.255.0 192.168.1.1 IF 13
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: " Now click on the **Advanced...** button to open the **Advanced TCP/IP Settings** window where we will make a few more changes".

> Warnings or important notes appear in a box like this.

> Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
DirectAccess Server Best Practices

In this chapter we are going to take a step-by-step approach in the preparation of your Windows Server 2012 Remote Access servers for use with DirectAccess. By walking through the process of preparing your servers, we will have ample opportunity to discuss what the changes and options that you are choosing actually mean, and give a little insight as to whether or not you really want to choose them. There are numerous ways in which DirectAccess in Server 2012 can be implemented, and not all the options are created equally. We'll discuss which options are the best in terms of security, and I'll describe the steps to take to make sure your environment is running as efficiently and securely as possible. The topics covered in this chapter are relevant to the actual server itself, and not necessarily DirectAccess environmental practices, as we will discuss those topics in *Chapter 2*, *DirectAccess Environmental Best Practices*.

Here's the layout of what we are going to look at:

- Preparing your Remote Access servers for DirectAccess
- NIC configuration
- NIC binding
- MAC address spoofing for virtual machines
- Adding static routes
- Hostname and domain membership
- Time for certificates
- Adding the roles
- Don't use the Getting Started Wizard!
- Security hardening the server

Preparing your Remote Access servers for DirectAccess

We are first going to walk through some standard operating procedures that you will want to take on every one of your Windows Server 2012 servers that you are planning to turn into Remote Access / DirectAccess servers. Whether working on the first DirectAccess server in your entire environment, or preparing a second, third, or eighth node that will be joined to an existing DirectAccess load-balanced array, follow these steps to ensure those machines meet the requirements, and that you aren't going to run into messages or have to backtrack and adjust settings after running through the wizards to activate DirectAccess.

NIC configuration

The vast majority of DirectAccess implementations will be of the two-leg fashion, with a **Network Interface Card** (**NIC**) for the external network, and another NIC for the internal network. This makes perfect sense, because this is your gateway into the corporate network from the computers in the wild; therefore, to most it is viewed as an edge device, and having separation of internal and external networks is a common network security best practice. So, just make sure my server has two network cards, plug them into the right switches, and configure IP addresses like I do on my desktop, right? No. In the Windows world, you need to take great care when defining your networking topology, particularly with the default gateway setting. If there is one thing that you can take away from this section of the book, it is this: **the default gateway setting is only defined on the external NIC**. This means that we will have to do some manual work to make sure that the server knows how to contact all the resources it may need to contact, but we'll get to that in a few minutes. First, let's take a look at the NIC configuration settings you will want in place to adhere to best practices. Whether you are new to DirectAccess or want to review an existing configuration that has been running for months, these steps are all relevant to you.

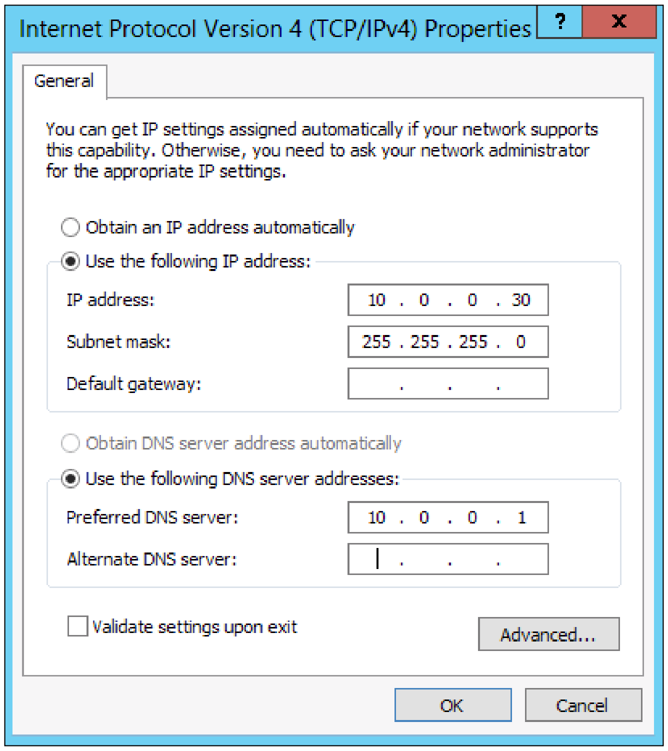
Configuring internal NIC

Let us go ahead and configure our internal interface first, because let's face it, you're already sick of standing in the elevated decibel level of the server room. Once you have the internal card configured with an IP address, assuming you have enabled RDP as on any other server of course, chances are you can run back to the comfort of your own desk and finish the job from there. Keep in mind that because we will NOT be defining a default gateway address on this NIC, you may not have access to this server over the network after simply defining an IP address.

You may have to add some routes before you can get to it from your desk, in which case you'll have to bunker down and endure console access for a little while longer, until we get through the section here about defining your static routes. In any case, before long you can stop sniffing the argon gas.

Name your NICs intuitively. If you rename your NICs to common-sense conventions like **Internal** and **External** instead of Local Area Connection 435, it will save you time during the wizards when you are defining which interface is which.

Open the **Properties** window of the internal NIC, and head into the **Internet Protocol Version 4 (TCP/IPv4) Properties** section, the same place where you would define an IP address on any computer. If you are using IPv6 inside your network, then you will be defining that instead, or in addition to IPv4 if you are running dual-stack. And if this is you, I applaud you immensely, because you are one of the very few, in my experience, who have taken this venture into IPv6 on your internal network. I say this only to point out that the overwhelming majority of internal networks are still IPv4, and so my examples and screenshots will be reflecting that scenario during the course of this book.



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

● Use the following IP address:

IP address: 10 . 0 . 0 . 30

Subnet mask: 255 . 255 . 255 . 0

Default gateway: . . .

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

Preferred DNS server: 10 . 0 . 0 . 1

Alternate DNS server: . . .

☐ Validate settings upon exit

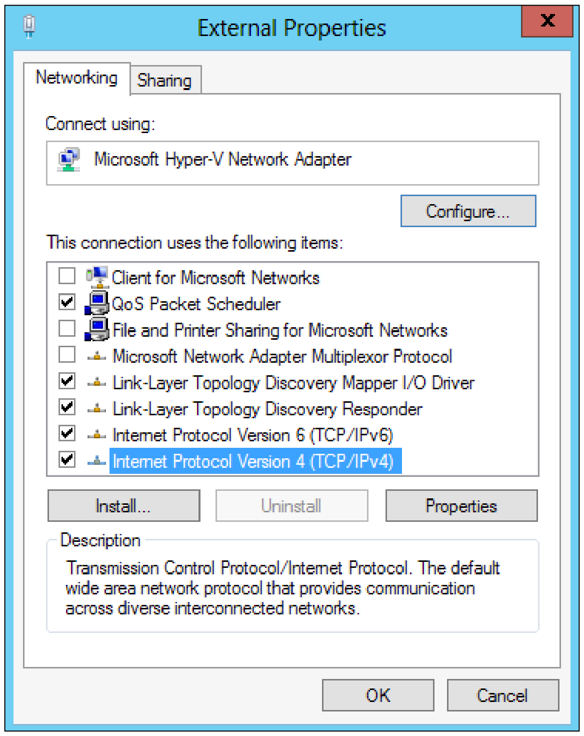
The fields in the previous window are as follows:

- **IP address**: You, of course, need to assign your internal IP here.
- **Subnet mask**: Please provide the appropriate mask; make sure it's accurate!
- **Default gateway**: Leave this field blank. We will not be defining an internal gateway.
- DNS servers: Yes, do provide your internal DNS server(s) here.

Configuring external NIC

Now we head over to the same properties page on the external NIC, but before we start defining IP addresses, there are a couple of things we can uncheck as they are not necessary, and unbinding anything that is not necessary only helps to improve the security and performance of the solution.

1. On your external NIC properties page, try to mirror the following screenshot:



Connect using:

Microsoft Hyper-V Network Adapter

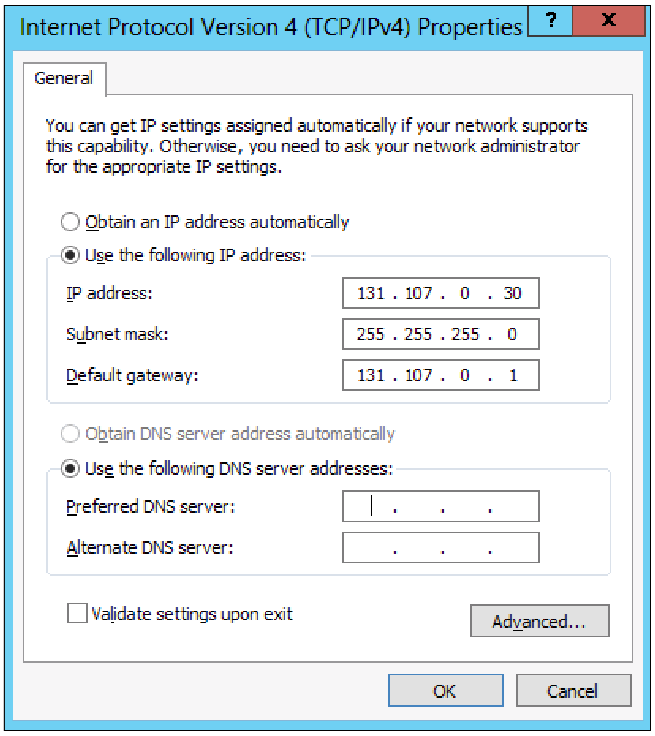
This connection uses the following items:

- [ ] Client for Microsoft Networks
- [✓] QoS Packet Scheduler
- [ ] File and Printer Sharing for Microsoft Networks
- [ ] Microsoft Network Adapter Multiplexor Protocol
- [✓] Link-Layer Topology Discovery Mapper I/O Driver
- [✓] Link-Layer Topology Discovery Responder
- [✓] Internet Protocol Version 6 (TCP/IPv6)
- [✓] Internet Protocol Version 4 (TCP/IPv4)

Description

Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

2. Mark the following checkboxes from the previous screenshot as shown:
   - **Client for Microsoft Networks**: Uncheck this box
   - **File and Printer Sharing for Microsoft Networks**: Uncheck this box
3. After unchecking these couple of boxes, head into the TCP/IPv4 properties, and enter your external IP address information.



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

Obtain an IP address automatically [✓ Use the following IP address:]

IP address:	131 . 107 . 0 . 30
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	131 . 107 . 0 . 1

Obtain DNS server address automatically [✓ Use the following DNS server addresses:]

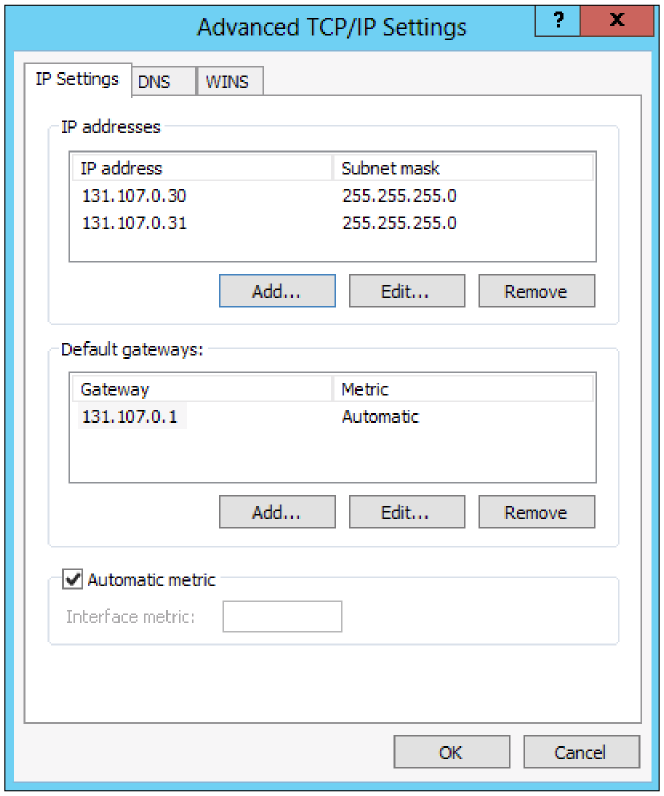
Preferred DNS server:	. . .
Alternate DNS server:	. . .

☐ Validate settings upon exit

4. The fields in this window are as follows:
   - **IP address**: Assign your primary external address here.
   - **Subnet mask**: Please provide the appropriate mask; make sure it's accurate!
   - **Default gateway**: Yes, we do need the public/external gateway address defined here. Take special care to ensure this too is accurate.
   - DNS servers: No, we do not define the DNS servers for the external connection, only the internal.

> The following steps will reflect the most common and recommended implementation path for DirectAccess, utilizing two public IP addresses on the external interface. Other installation scenarios such as single public IP or single private IP, or even for a single NIC implementation, would not require a second IP address to be added here.

5. Now click on the **Advanced...** button to open the **Advanced TCP/IP Settings** window where we will make a few more changes.
6. Assuming that you are taking the install path requiring two public IP addresses, go ahead and click on the **Add...** button to input your second **IP address** and **Subnet mask**. You are not required to input another default gateway, only the IP and mask.

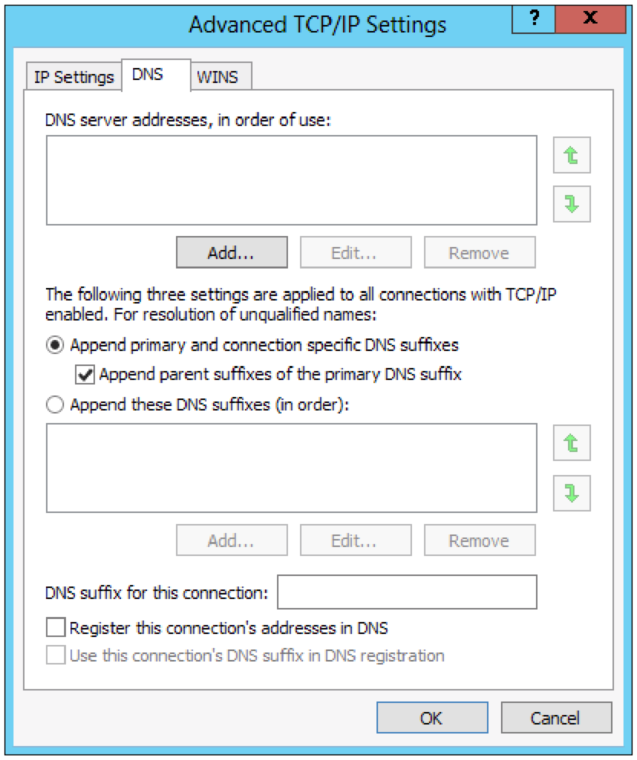


IP address	Subnet mask
131.107.0.30	255.255.255.0
131.107.0.31	255.255.255.0

Gateway	Metric
131.107.0.1	Automatic

☑ Automatic metric

7. Now head over to the **DNS** tab and uncheck the **Register this connection's addresses in DNS** checkbox.



Advanced TCP/IP Settings

IP Settings | DNS | WINS

DNS server addresses, in order of use:

The following three settings are applied to all connections with TCP/IP enabled. For resolution of unqualified names:

[✓ Append primary and connection specific DNS suffixes]

[✓ Append parent suffixes of the primary DNS suffix]

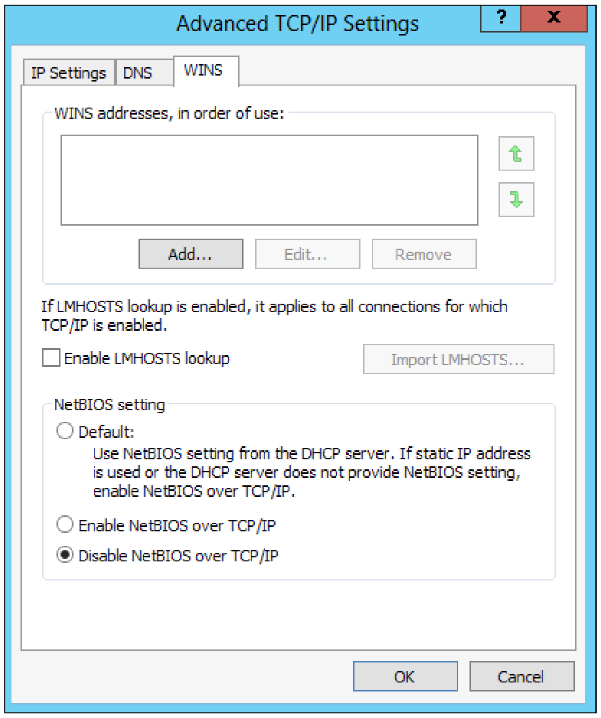
Append these DNS suffixes (in order):

DNS suffix for this connection:

☐ Register this connection's addresses in DNS

☐ Use this connection's DNS suffix in DNS registration

8. Finally, move one more tab over to the **WINS** tab and here you want to uncheck **Enable LMHOSTS lookup** and set the radio button for **Disable NetBIOS over TCP/IP**.



WINS addresses, in order of use:

If LMHOSTS lookup is enabled, it applies to all connections for which TCP/IP is enabled.

[ ] Enable LMHOSTS lookup

NetBIOS setting

○ Default:
Use NetBIOS setting from the DHCP server. If static IP address is used or the DHCP server does not provide NetBIOS setting, enable NetBIOS over TCP/IP.

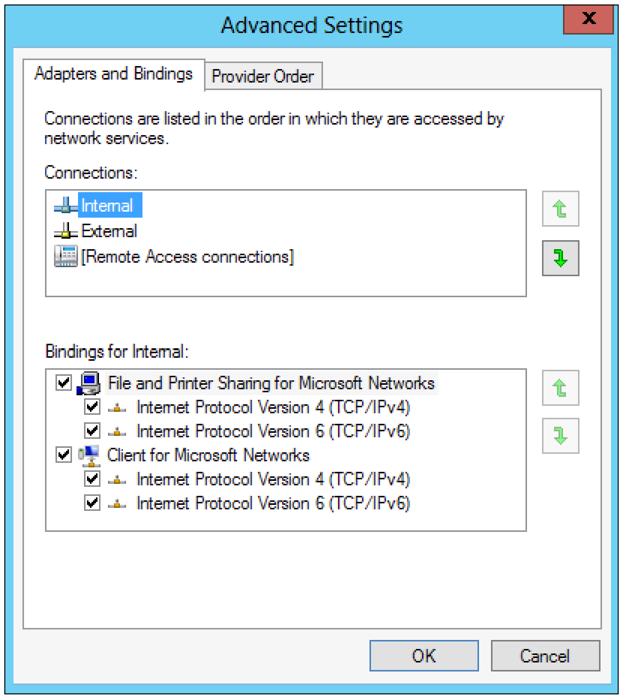
○ Enable NetBIOS over TCP/IP

[✓ Disable NetBIOS over TCP/IP]

Now you can click on **OK** three times to bring you back to the **Network Connections** window where you are looking at your NICs. Before you leave this screen, you want to make sure and set your NIC binding appropriately.

NIC binding

To set this, while you are in the **Network Connections** screen, press the *Alt* key on your keyboard to bring up the menus on top of the window. Then head over to the **Advanced** menu and click on **Advanced Settings…** This will open the **Adapters and Bindings** section, and here we want to make sure that your Internal NIC is listed first, and that your External NIC is listed second. So, click on the names of the adapters, and use the arrows on the right side of the screen to move them up and down. If you have more NICs in your server, we don't necessarily care about the rest, as long as Internal is first, and External is second.


Advanced Settings

Adapters and Bindings | Provider Order

Connections are listed in the order in which they are accessed by network services.

Connections:

- Internal
- External
- [Remote Access connections]

Bindings for Internal:

- [✓] File and Printer Sharing for Microsoft Networks
  - [✓] Internet Protocol Version 4 (TCP/IPv4)
  - [✓] Internet Protocol Version 6 (TCP/IPv6)
- [✓] Client for Microsoft Networks
  - [✓] Internet Protocol Version 4 (TCP/IPv4)
  - [✓] Internet Protocol Version 6 (TCP/IPv6)

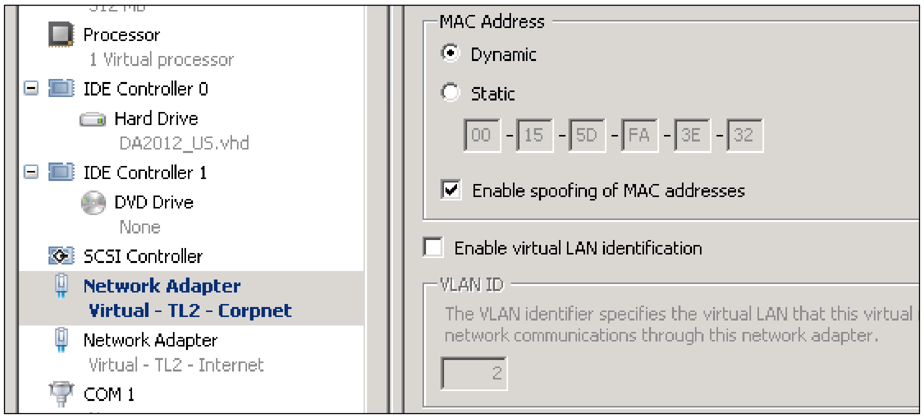
In fact, personally, I always disable any NICs on the system that are not in use by DirectAccess. Many folks come into preparing their server for DirectAccess thinking of it like a firewall, and on a firewall having too many NICs is always better than not having enough, but unfortunately DirectAccess cannot take advantage of more than two NICs. Rather, DirectAccess cannot take advantage of more than two "legs". You can do NIC teaming, but you still have the limitation of only working with one Internal and one External leg. So, determine which is your Internal and which is your External, name them, IP them, and bind them appropriately, and then you can go ahead and simply disable all of the rest of your NICs. This is not required, but I consider it a best practice in accordance with the idea of disabling anything that is not needed on a networking device.

Many Network Cards now come with a feature named **Receive Side Scaling** (**RSS**). This setting is often beneficial to a DirectAccess server and should be enabled on your NICs. It is likely to be already enabled by default, but if you want to check, you can head back into the NIC properties and click on the **Configure…** button. Head into the **Advanced** tab and look for the RSS settings that are particular to your NIC.

MAC address spoofing for virtual machines

If your DirectAccess server is a virtual machine, which doesn't necessarily line up with my idea of a best practice in any way, but I understand that many folks do it; make sure to set your NICs to allow MAC address spoofing. This will be particularly important if and when you decide to create any kind of arrays or load-balanced clusters, but I recommend always making this change right in the beginning, so that you are prepared for those situations and don't have to take troubleshooting steps down the road. To set this setting in Hyper-V, go into your Hyper-V Manager, right-click on your DirectAccess virtual machine, and click on **Settings…**.

Find your network adapter listed on the left and click on the **+** symbol next to it to drop down some additional options. Click on **Advanced Features**, and then over on the right, check the checkbox for **Enable spoofing of MAC addresses**. Depending on your version of Hyper-V, the setting might be in a slightly different section of the network adapter's properties. For example, here it is on a Server 2008 R2 Hyper-V server.



[✓ Dynamic] Static

[✓] Enable spoofing of MAC addresses

[ ] Enable virtual LAN identification

The VLAN identifier specifies the virtual LAN that this virtual network communications through this network adapter.

You have to check this setting for both the network adapters that are being used by DirectAccess. Also, keep in mind that changing this setting requires the virtual machine to be turned off. If your MAC address spoofing option is grayed out, shut down the virtual machine and then check it again.

Whew, we're finally finished with all of the NIC configurations. Seems like a lot of text just to make sure something as simple as network settings was configured properly, but it is absolutely critical to make sure you have a solid networking baseline before you try to configure DirectAccess. If you do not, if any of the settings listed are not correct, if there is an incorrect subnet mask listed somewhere, if you have put a default gateway on the internal NIC, and the list goes on and on…if network settings are not configured properly, you will run into error messages, or maybe worse no error message but strange client behavior that can't be explained. Incorrectly configured networking settings can also cause a DirectAccess server to "lose itself", resulting in the console hanging and your only recourse to be a complete server re-prep so that you can start over. Make sure your NICs are configured correctly!

Adding static routes

At this point, the astute among you are saying, "Wait a minute, we only put a default gateway on the External NIC, not on the Internal. My network is comprised of many internal subnets, and this server isn't going to be able to contact any of those subnets without a default gateway!" You are absolutely correct. Because we can only have one default gateway and it must go on the external interface, we have to define our internal network manually, through the use of the Windows routing table. Your server will automatically have access to resources that are in the same subnet that you are physically connected to, so if your IP address is 192.168.1.10 and your whole network is a flat 192.168.1.0/24, then there is nothing you have to do. The DirectAccess server will have access to everything in 192.168.1.x and you are all set. If, however, you have additional subnets, 192.168.2.0/24 for example, then at this point they are not contactable from this server and we need to make it so. You do this through the use of **route** commands, issued from either the Command Prompt or the PowerShell interface. I find that most folks are more familiar with Command Prompt, so let's use that to make our changes.

First we'll start with the example listed previously. Say my DirectAccess server is 192.168.1.10, but I have file servers that are sitting in 192.168.2.0/24, and those file servers must be contactable by the DirectAccess client computers. All we have to do is run a simple command on your DirectAccess server to make this happen. Here is an example of the syntax of that command:

```
Route add -p <subnet> mask <subnet mask> <gateway> IF <Interface ID>
```

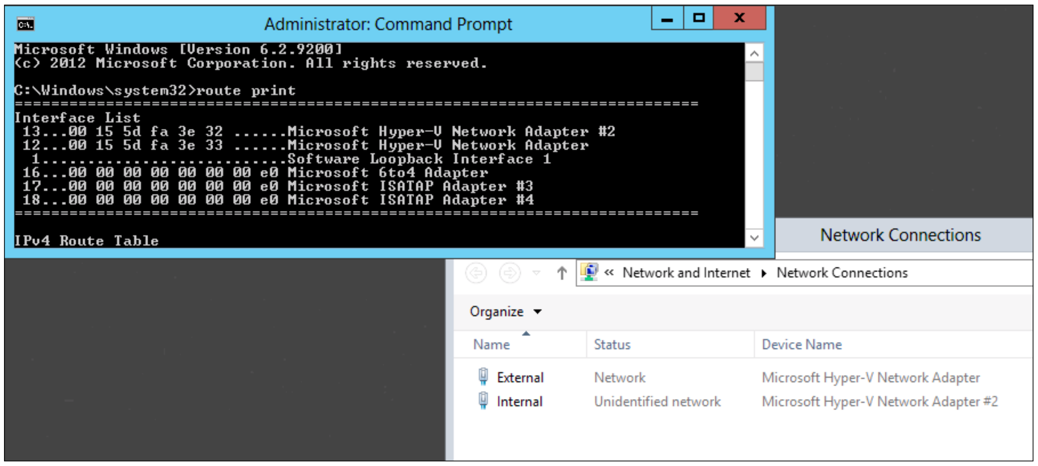
The various components of this command are as follows:

- **-p**: This makes the command persistent. Without –p, the next time the server restarts, the route would be lost.
- **<subnet>**: This is the subnet ID you are adding, such as 192.168.2.0.

- **<subnet mask>**: This is the mask for that subnet, such as 255.255.255.0.
- **<gateway>**: This is the gateway of the subnet that you are currently plugged into, NOT the gateway of the subnet you are adding. Think of this as the "first hop" that you must cross in order to contact this new subnet. In our example, the gateway is 192.168.1.1.
- **<Interface ID>**: This is an identifier for our Internal NIC that we will discuss in the next paragraph.

To be able to formulate that command correctly, we first need to identify the Interface ID number of your Internal NIC. Since we have dual network cards in this server, it is important that we are applying these route statements to the internal card. There is a flag that we will set at the end of our route commands that binds our route to a particular card, and most of the time Windows does a good job of assigning it to the correct one without validating this **IF** number, but I have seen a few cases where it didn't, so I always specify it as a best practice.

To discover your Interface ID number for the Internal NIC, open both the **Network Connections** screen, and a command prompt, and type `route print`. If you scroll up to the very top of your route print, you will see each network interface that is on the system listed, with an IF number listed to the left of the name. This typically two-digit number is the IF number for the NIC, and we just need to figure out which one is Internal. That is where the **Network Connections** screen comes in. If you take a look at the full name of the Internal NIC, match it up with the full name listed for one of the NICs in the route print; there you have it. Let's say, for example, that your Internal NIC is named **Microsoft Hyper-V Network Adapter #2**, and in your route print you see **13...00 15 5d fa 3e 32 ......Microsoft Hyper-V Network Adapter #2** listed, as shown in the following screenshot:


Administrator: Command Prompt
Network Connections
« Network and Internet › Network Connections
Organize

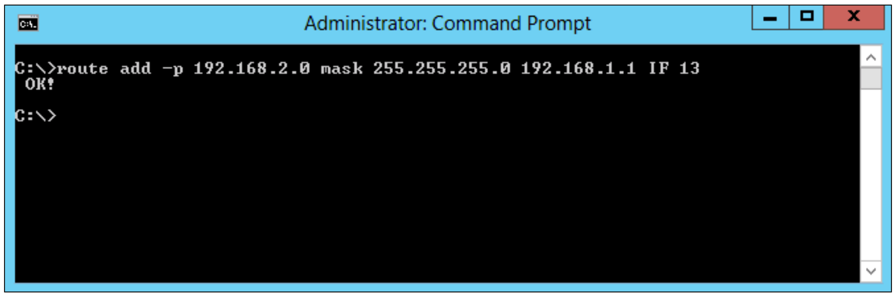
```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Windows\system32>route print
===========================================================================
Interface List
 13...00 15 5d fa 3e 32 ......Microsoft Hyper-V Network Adapter #2
 12...00 15 5d fa 3e 33 ......Microsoft Hyper-V Network Adapter
  1...........................Software Loopback Interface 1
 16...00 00 00 00 00 00 00 e0 Microsoft 6to4 Adapter
 17...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #3
 18...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #4
===========================================================================

IPv4 Route Table
```

Name	Status	Device Name
External	Network	Microsoft Hyper-V Network Adapter
Internal	Unidentified network	Microsoft Hyper-V Network Adapter #2

In this case, the IF number for your Internal NIC is 13, the number listed to the far left of that line. Taking that Interface ID number combined with the sample route statement above, let's go ahead and build the route statement that we would need to successfully grant access to the 192.168.2.0/24 subnet on your server by using the following command:

```
route add -p 192.168.2.0 mask 255.255.255.0 192.168.1.1 IF 13
```

If you have entered all of the information correctly, you should see the following **OK!** response:



```
C:\>route add -p 192.168.2.0 mask 255.255.255.0 192.168.1.1 IF 13
 OK!

C:\>
```

Now, before you start dreading the huge script that you might be thinking about creating to include the potentially hundreds of route statements you may need in your network, read this first. Depending on the layout of your network, it may be possible to include a much broader route statement and cover all of your subnets in one fell swoop. Building on our previous example, what if your DirectAccess server was 192.168.1.10, and you had many subnets, all of them in the 192.168.x.x range? You could cover all of these subnets and tell them to all flow through the Internal NIC with the following single command:

```
route add -p 192.168.0.0 mask 255.255.0.0 192.168.1.1 IF 13
```

Or even broader. And by the way, this is of course not only limited to subnets starting with 192.168. Another example I can give which I have encountered numerous times in different customer networks is the following one:

```
route add -p 10.0.0.0 mask 255.0.0.0 10.1.1.1 IF 13
```

Take care that you do not specify a route that is so broad that it encompasses the subnet of the External NIC. If you add a route to the Internal interface which includes the subnet for the External NIC, you will cause major confusion on the server and will almost certainly stop DirectAccess from working.

You should now have all the information you need to finalize your IP addressing and routing on your DirectAccess servers. These steps are necessary on each server. Just one more side note to add here; implementing DirectAccess in the single NIC configuration isn't something I see much in the wild, but in those cases you would not have to go through this process of adding routes. This is because in a single NIC configuration, you would be assigning a default gateway right on the single NIC that is in use, and that gateway is going to cover any routes that you may have to enter otherwise.

Hostname and domain membership

Now that our network traffic is flowing, we need to finalize a couple of other regular items on the DirectAccess server. First is setting the hostname. While this seems like a menial, regular task, don't take it lightly. It is recommended that once your hostname is set, it should not be changed in the future. So choose the name carefully, and choose a name that meets your naming standards. It is not recommended to change the hostname of a DirectAccess server, because there are items external to the server itself which are bound to that particular name, such as Group membership, **Group Policy Objects** (**GPOs**) filtering, and certificates. A change in the hostname of a DirectAccess server will result in a number of external factors needing to be changed, adjusted, or reissued, and there is a huge potential for problems. So all that to say—choose your name wisely and don't think you can name it DA-Test for now, and simply rename it later.

Once your name is set, it is time to join it to the domain. This is required for DirectAccess to work, as the solution is tightly integrated with Active Directory. You do not have to join it to the same domain as the rest of your internal network or the same domain as the DirectAccess client machines, but whatever domain you join it to must have a two-way trust to those domains, so that traffic can flow successfully between the DirectAccess server and the resources with which it is going to interact.

Prestage the computer account

I highly recommend prestaging the computer account for your DirectAccess server(s) in Active Directory before you join them to the domain. This is not required, but I recommend it because I have seen many cases where upon joining the domain, a DirectAccess server had some existing GPOs applied to it which disabled items in Windows that are necessary for DirectAccess to function. What I see most often are GPOs in place on the network which disable or make changes to the Windows Firewall, and if any of these policies get applied to your DirectAccess servers, it will certainly interfere with operability.

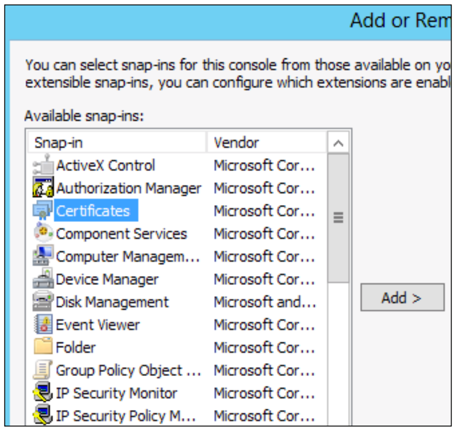
Try your best to make sure that no existing polices get applied to the servers at all. It is best to create a separate **Organizational Unit** (**OU**) for them to reside in, which blocks the inheritance of existing policies. In the end, there are going to be policies that need to apply to them, the actual DirectAccess Server policy for example, but try to keep them as clean as possible from changes. Once you have DirectAccess connectivity established and working, you can try applying your policies one at a time if you so choose, but keep in mind that if a GPO gets applied and changes are made, then simply removing the device from that GPO's filtering doesn't always reverse the settings that were changed. It is possible that you could break the DirectAccess server to the point that the quickest resolution is to re-prep the server and start over, so tread lightly here.

Time for certificates

Your server is almost ready to service DirectAccess connections! The last thing we want to do before adding the Remote Access role is to put all of our certificates into place on the server. We will talk more extensively about certificates and what options are available to us in *Chapter 2*, *DirectAccess Environmental Best Practices*, but in almost every implementation there are two certificates with which you want to be concerned at this point.

Installing the IP-HTTPS SSL certificate

For the purposes of this book, we're not going to talk much about what IP-HTTPS is, but the key for this section is that we need an SSL certificate installed onto the DirectAccess server that is going to validate the connections coming in. Any time that you want to view, add, or change certificates on a DirectAccess server, you are best to do so using the Certificates snap-in for the Microsoft Management Console. Open the console on your DirectAccess server, and navigate to **File** | **Add/Remove Snap-in...**. Then choose the **Certificates** snap-in.

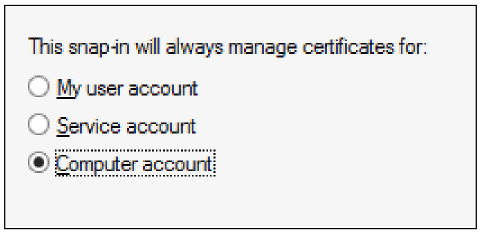

Add or Rem

You can select snap-ins for this console from those available on yo
extensible snap-ins, you can configure which extensions are enabl

Available snap-ins:

Snap-in	Vendor
ActiveX Control	Microsoft Cor...
Authorization Manager	Microsoft Cor...
Certificates	Microsoft Cor...
Component Services	Microsoft Cor...
Computer Managem...	Microsoft Cor...
Device Manager	Microsoft Cor...
Disk Management	Microsoft and...
Event Viewer	Microsoft Cor...
Folder	Microsoft Cor...
Group Policy Object ...	Microsoft Cor...
IP Security Monitor	Microsoft Cor...
IP Security Policy M...	Microsoft Cor...

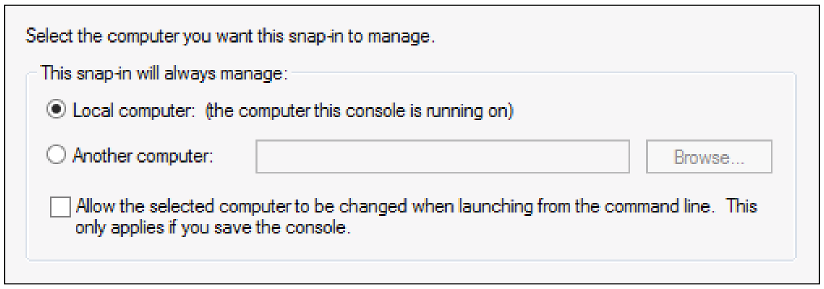
When you click on the **Add** button, you will be prompted to choose which certificate store you want to manage. We always want to choose **Computer account** when we are dealing with DirectAccess certificates.



This snap-in will always manage certificates for:

- My user account
- Service account
- [✓ Computer account]

And on the next screen, choose **Local computer**.



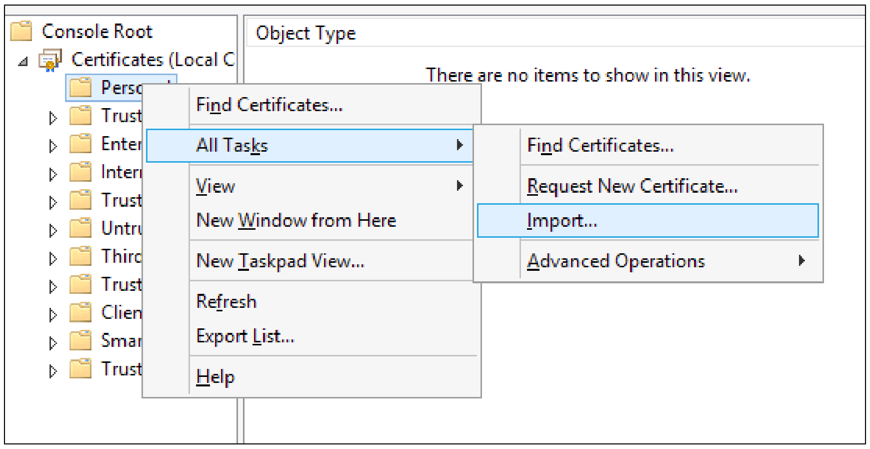
Select the computer you want this snap-in to manage.

This snap-in will always manage:

- [✓ Local computer: (the computer this console is running on)]
- Another computer: Browse...

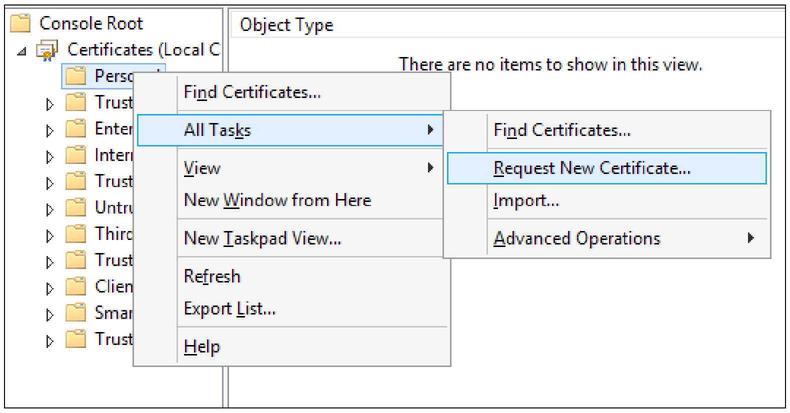
Allow the selected computer to be changed when launching from the command line. This only applies if you save the console.

Now, if you navigate to **Certificates** | **Personal**, right-click and choose **All Tasks** | **Import...** and finish out the wizard to import the SSL certificate that you have acquired for the purposes of IP-HTTPS.

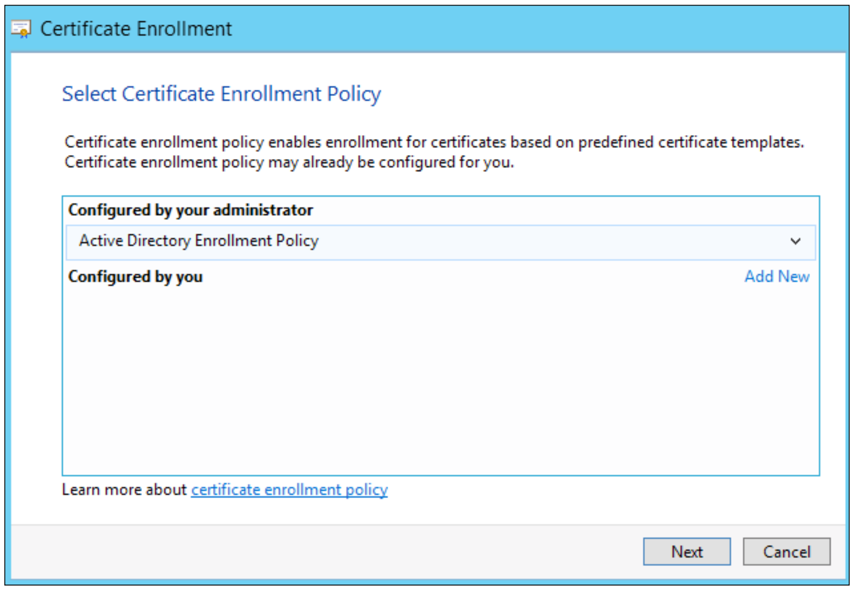


Installing the IPsec machine certificate

The other certificate that we want to make sure exists in this same certificate store on the DirectAccess server, in almost every DirectAccess implementation scenario, is a machine certificate that has been issued by your internal **Certification Authority** (**CA**) server. Many companies already have something called autoenrollment enabled in their network which automatically issues certificates to machines as soon as they join the domain. If this is the case, you will already see a (or many) certificate(s) listed inside the **Personal** certificate store. If this certificate was issued from the internal CA server and the subject name of the certificate matches the FQDN of the DirectAccess server, this certificate may work for IPsec authentication. You can take a look at the next chapter of this book for further details on what criteria the IPsec certificate needs to meet to be successful for DirectAccess. Otherwise, for this example, we will assume that you do not have an IPsec certificate already assigned to your server, and we will walk through the process of requesting one from your internal **Public Key Infrastructure** (**PKI**). Right-click on the **Personal** certificate store again, but this time navigate to **All Tasks** | **Request New Certificate...**.



Click on **Next**, and then click on **Next** again on the screen that shows **Active Directory Enrollment Policy**. Nothing to change or adjust on this screen, simply click on **Next**.



Select Certificate Enrollment Policy

Certificate enrollment policy enables enrollment for certificates based on predefined certificate templates. Certificate enrollment policy may already be configured for you.

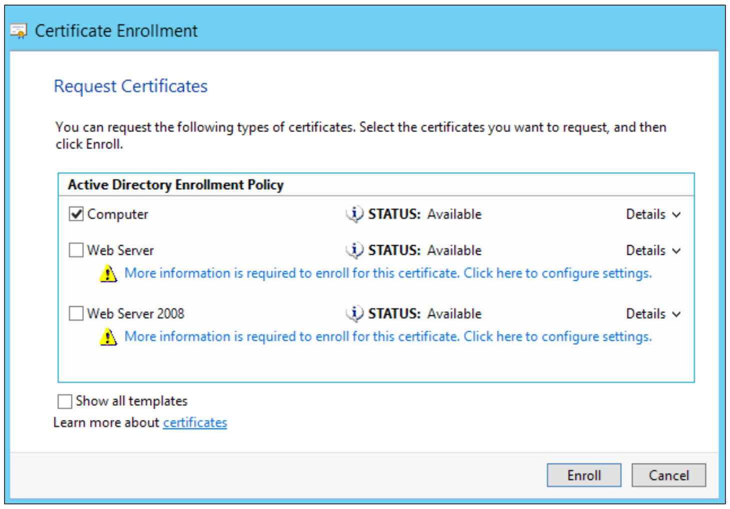
Configured by your administrator

Active Directory Enrollment Policy

Configured by you Add New

Learn more about certificate enrollment policy

This will poll your internal PKI for any certificate templates that are available to be issued. If your CA server is setup properly, you will see one or more options available to select, and hopefully one of these options is named **Computer**.



Request Certificates

You can request the following types of certificates. Select the certificates you want to request, and then click Enroll.

Active Directory Enrollment Policy

☑ Computer — **STATUS:** Available — Details

☐ Web Server — **STATUS:** Available — Details
More information is required to enroll for this certificate. Click here to configure settings.

☐ Web Server 2008 — **STATUS:** Available — Details
More information is required to enroll for this certificate. Click here to configure settings.

☐ Show all templates

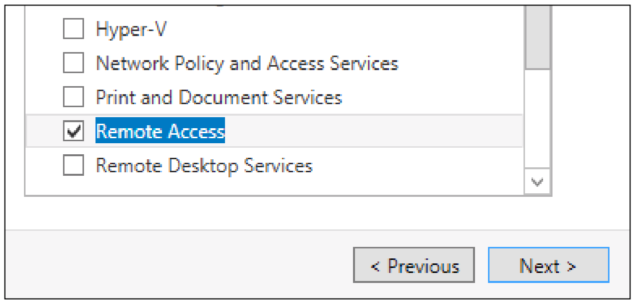
Learn more about certificates

This is a predefined template that exists in Windows CA, and meets all the requirements for a successful IPsec authentication certificate to be used with DirectAccess. You may have also chosen to create a custom template on the CA server that is going to be used specifically for DirectAccess, as detailed in the certificate details section in *Chapter 2*, *DirectAccess Environmental Best Practices*, and if that is the case, then you would have that option available to you as well for issuance. Either way, simply select the certificate template from the list for which you would like to request a certificate, click on **Next**, and you will be issued a machine certificate from the internal CA server onto your DirectAccess server, and this certificate will show up in the **Personal** certificate store.

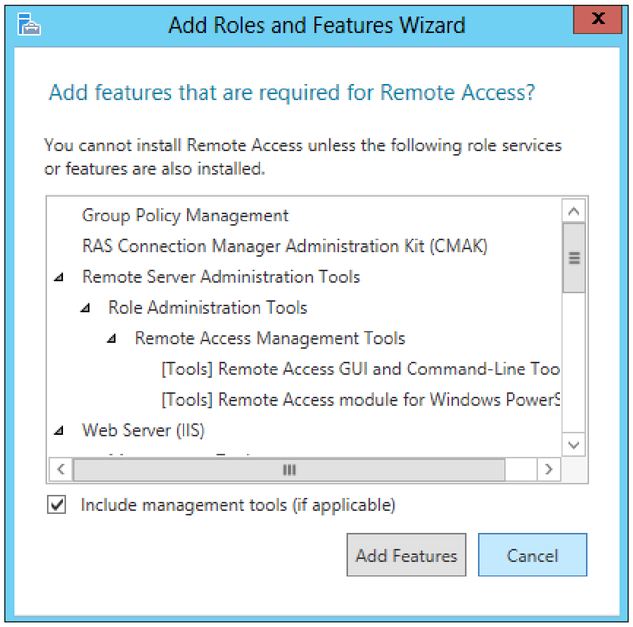
Adding the roles

Now that we have all of our settings and prerequisites in place on the DirectAccess server, the last step before we can get into the actual configuration is adding the Remote Access role, and possibly the Network Load Balancing feature, depending on your plan for implementation. To do this, as with any other role or feature, simply launch Server Manager if not already running, and click on the **Add roles and features** link from inside the dashboard. Click on **Next**, then click on **Next** again choosing the default option for role-based or feature-based installation, and click on **Next** once more on the screen showing your server name selected in the list.

Click on the **Remote Access** role and click on **Next**.


☐ Hyper-V
☐ Network Policy and Access Services
☐ Print and Document Services
☑ Remote Access
☐ Remote Desktop Services

You will now be prompted with a screen that shows some additional options that need to be enabled to support the Remote Access role. Go ahead and click on the **Add Features** button to continue.



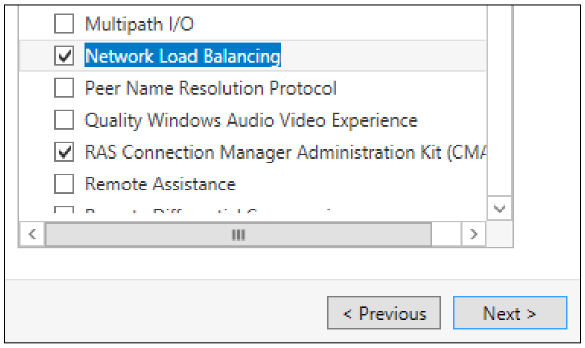
Add features that are required for Remote Access?

You cannot install Remote Access unless the following role services or features are also installed.

Group Policy Management
RAS Connection Manager Administration Kit (CMAK)
Remote Server Administration Tools
Role Administration Tools
Remote Access Management Tools
[Tools] Remote Access GUI and Command-Line Too
[Tools] Remote Access module for Windows PowerS
Web Server (IIS)

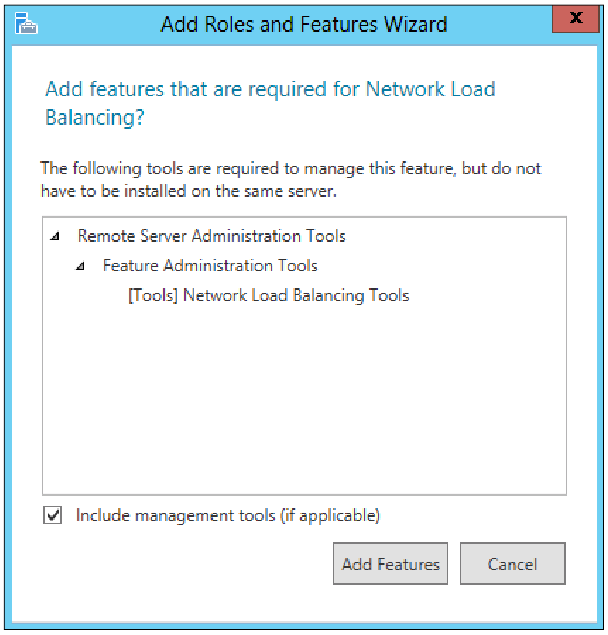
☑ Include management tools (if applicable)

Our next screen is for adding features to the server, and we may or may not have to do anything on this screen. If this is your one-and-only DirectAccess server, and you don't plan to ever have more, go ahead and simply click on **Next**. If you are interested in creating an array, or a cluster of DirectAccess servers in the future, or if this server is going to be an additional node to an existing array or cluster, then make sure to select the **Network Load Balancing** feature from this list before clicking on **Next**.



☐ Multipath I/O
☑ Network Load Balancing
☐ Peer Name Resolution Protocol
☐ Quality Windows Audio Video Experience
☑ RAS Connection Manager Administration Kit (CMA
☐ Remote Assistance

From here you will again be prompted that there are some additional items that need to be enabled to support this **Network Load Balancing** feature, go ahead and accept that screen to continue.


Add Roles and Features Wizard

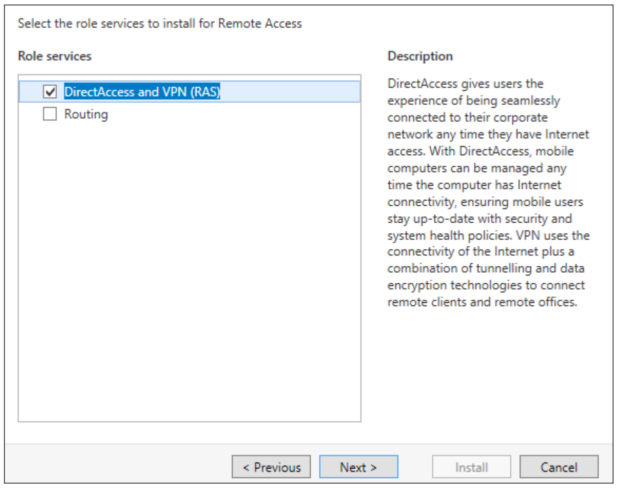
Add features that are required for Network Load Balancing?

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

- Remote Server Administration Tools
  - Feature Administration Tools
    - [Tools] Network Load Balancing Tools

☑ Include management tools (if applicable)

After clicking on **Next** a couple more times, you will be presented with an option on whether you want to utilize **DirectAccess and VPN (RAS)**, or **Routing**, or both. In my opinion, a DirectAccess server should be a remote access platform and nothing else, so let's stick with the defaults, and choose only the **DirectAccess and VPN (RAS)** option.


Select the role services to install for Remote Access

Role services

☑ DirectAccess and VPN (RAS)
☐ Routing

Description

DirectAccess gives users the experience of being seamlessly connected to their corporate network any time they have Internet access. With DirectAccess, mobile computers can be managed any time the computer has Internet connectivity, ensuring mobile users stay up-to-date with security and system health policies. VPN uses the connectivity of the Internet plus a combination of tunnelling and data encryption technologies to connect remote clients and remote offices.

Now simply finish out the wizard using all the default settings, and your server is officially ready for use with DirectAccess!

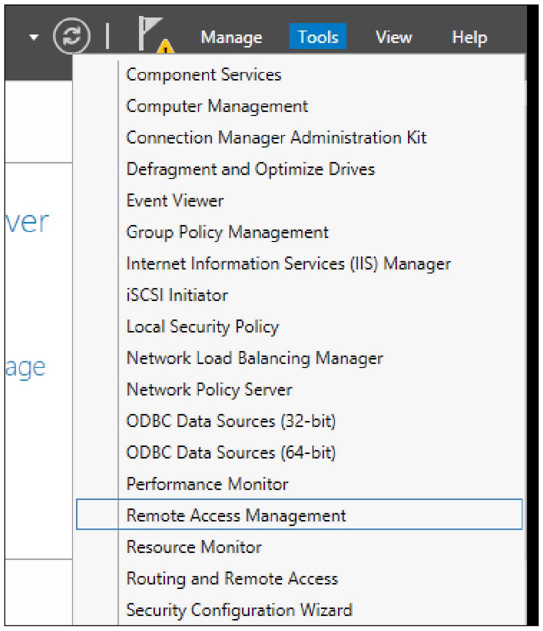
This ends the section of steps that you want to take on each of your servers to prepare them for use with DirectAccess. After adding the roles, you are now ready to either start actual DirectAccess configuration if this is your primary server, or ready to add this server to your array if this is an additional server that you are adding to an existing DirectAccess environment.

Don't use the Getting Started Wizard!

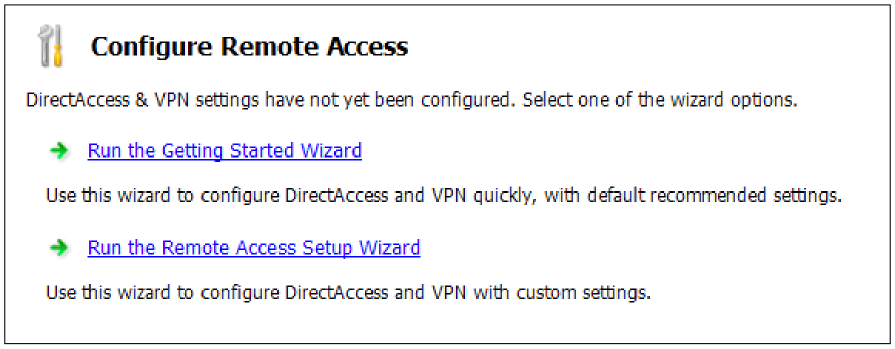
For the purposes of this book, I do not have plans to walk through all of the configuration wizards and explain each and every step that will be taken while walking through those wizards. What I do want to accomplish is to take a minute and point out one critical note. Many of the DirectAccess "walk-through" or "Test Lab Guide" documents that exist will tell you at this point in the process to run the Remote Access Management console and launch that great **Getting Started Wizard**. You know, the one where you can "install DirectAccess in 3 clicks!" Please don't do this! I understand why they included this option, so that literally anybody with a mouse could get some semblance of DirectAccess up and running, but running this wizard follows zero best practices, and I would hope that anyone reading this guide about best practices in your DirectAccess environment would have no interest in taking shortcuts during your install.

Running the full Remote Access Setup Wizard

I have spoken with many new DirectAccess administrators who didn't actually know they had a choice. It's pretty easy to glaze over the option and just follow whatever quick-start guide you are using and choose the Getting Started Wizard. So, I want to point out the way to launch the real wizards instead. After you add the Remote Access role, your next step is configuring DirectAccess. To do that, while you are still inside Server Manager, you can navigate to the **Tools** menu and choose **Remote Access Management** from the menu.


Manage
Tools
View
Help
ver
age
Component Services
Computer Management
Connection Manager Administration Kit
Defragment and Optimize Drives
Event Viewer
Group Policy Management
Internet Information Services (IIS) Manager
iSCSI Initiator
Local Security Policy
Network Load Balancing Manager
Network Policy Server
ODBC Data Sources (32-bit)
ODBC Data Sources (64-bit)
Performance Monitor
Remote Access Management
Resource Monitor
Routing and Remote Access
Security Configuration Wizard

The first screen you encounter here is your fork in the road. Clicking the top link here obviously launches the Getting Started Wizard. The second link listed under it, **Run the Remote Access Setup Wizard**, is the link that takes you into the full configuration wizards, and is absolutely the way that you want to go.



Configure Remote Access

DirectAccess & VPN settings have not yet been configured. Select one of the wizard options.

Run the Getting Started Wizard

Use this wizard to configure DirectAccess and VPN quickly, with default recommended settings.

Run the Remote Access Setup Wizard

Use this wizard to configure DirectAccess and VPN with custom settings.

Reasons not to use the Getting Started Wizard

I cannot tell you to stay away from the Getting Started Wizard without backing that up with a little bit of data, so let's talk about some of the reasons that I recommend handling this wizard with a ten-foot pole.

Self-signed certificates

Hopefully, now that you have read the beginning of this chapter, you know that you should input your certificates onto the server before you even add the roles. Unfortunately, most DirectAccess admins are not aware of this, and so the roles get added and the wizards run before the certificate is in place. When you run the Getting Started Wizard, if your certificate for IP-HTTPS is in place, it will recognize and use it, but if you do not have a valid certificate in place, it will generate and use a self-signed certificate for this purpose. The wizard will also generate and use a self-signed certificate for the **Network Location Server** (**NLS**) website. Using self-signed certificates is fine for a Proof-of-Concept or a Test Lab, but they are obviously a very bad practice for a production environment. Using a self-signed certificate means that your DirectAccess server can be easily spoofed, and the old 1024-bit key length used by self-signed certificates is no longer considered to be strong enough.

Self-hosted NLS

As we will discuss more in *Chapter 3*, *Configuring Manage Out to DirectAccess Clients*, a DirectAccess environmental best practice is to host the NLS website externally to the DirectAccess server. When you use the Getting Started Wizard, for the sake of saving mouse clicks, it assumes that you want to host the NLS website on the DirectAccess server. It also issues a self-signed certificate for this site.

Disables Teredo

Running the Getting Started Wizard also assumes that you are not interested in using any transition protocol other than IP-HTTPS, and disables them. We will talk some more shortly about the reasons that you want Teredo available to you if possible in your environment, so this again works against best practices in an effort to make implementation as automated as possible.

Applies client policy to the domain computers group

Yikes! If you have ever run through the real DirectAccess wizards, or have been through the UAG DirectAccess wizards, you know that the client-side GPO settings are filtered out to only the actual DirectAccess client computers by way of Active Directory group membership. During the wizards, we define the group or groups in Active Directory that are going to contain our DirectAccess client computers, and the wizard defines security filtering on the client-side GPO, so that those settings only come down to the actual DirectAccess computers. This is a great idea! You, of course, don't want a bunch of remote access connectivity settings distributing themselves around your internal desktop computers, or worse yet servers in your network.

The Getting Started Wizard has the potential to do just that. When you click on this mini-wizard, it flags the GPO to apply to all domain computers. Now, it does set a WMI filter on this so that it only applies to mobile computers as defined in the Windows WMI filter, so chances are that if you have already run the Getting Started Wizard in your network, at least these settings aren't running rampant, but those WMI filters are far from 100 percent accurate, and I just dread thinking about all of those networks out there where the link in their DirectAccess GPO right now says "Domain Computers".

No advanced choices

This one is pretty obvious, but I'll state it nonetheless. If you run the Getting Started Wizard, you won't encounter all of the optional settings that DirectAccess has available, so you may miss out on some advanced implementation of which you want to make use. If you have already implemented DirectAccess by using the Getting Started Wizard and are in production and don't want to start over on your configuration, you can still make adjustments to your settings and overcome the limitations that were put into place by the wizard. Inside Remote Access Management, you can head into the Configuration section and gain access to the individual steps of the real wizard here to make adjustments after the fact. Keep in mind though, that you do not have the option to change GPOs afterwards, and that running the Getting Started Wizard disables Teredo at a lower level that cannot be overcome by the wizards. You must use PowerShell to enable it, and that process is detailed in the last chapter of this book, when we discuss some specific troubleshooting scenarios.

Security hardening the server

Most network administrators view the DirectAccess server, rightfully so, as an "edge" device. Because this device is going to sit on the edge, or so close to it, there is a lot of sense in locking down security in whatever ways you can. You could, of course, take the route of purchasing a prebuilt security appliance for running DirectAccess instead of using your own server. Full disclaimer here; I do work for one of the appliance manufacturers who builds and installs DirectAccess Concentrators every day, but I bring this up only because it is a legitimately secure and performant way to implement DirectAccess in any environment. There are substantial advantages that appliances hold over regular servers, but there is, of course, cost also involved with the purchase of the appliance.

When configuring your own server for DirectAccess, try to think of this device as a single-purpose server. Don't put other roles or features or install applications other than what are necessary for DirectAccess, and consider disabling services that are not going to be needed. I can't create a comprehensive list of exactly what should be buttoned down, because it could be different for each network, but run through the list of items that are installed or running on the device and try to decide whether or not they are necessary to be running on this server. A security best practice that goes back to the beginning of time is to lower your potential threat footprint by disabling services that are unnecessary, and that is a good practice here as well.

It is recommended to install antivirus onto the DirectAccess server, as with any other server, but it is important that you do not install an antivirus that includes its own firewall, which may manipulate or squash the Windows Firewall. **Windows Firewall with Advanced Security**, or **WFAS**, that is built into Windows is essential to the operability of DirectAccess and cannot be turned off. WFAS needs to be active on your DirectAccess server.

Windows Firewall is much more comprehensive than it used to be, and can be used for making additional lockdowns on your device as well. For example, in the instance where you have the DirectAccess server installed directly onto the public Internet, by default when you enable Remote Access (RDP) to the server, it enables for all firewall profiles. This means that port 3389 is opened to the outside world as well as the inside. It is a common best practice to either adjust the RDP rule in WFAS, or to create a new **deny** rule in WFAS that blocks port 3389 for the **Public** and **Private** firewall profiles, which would be the potential active profiles on the external connection. This way you would only be allowing RDP access to the server from the **Domain** profile, or inside of your network. You could also increase the security of RDP access in general by requiring **Network Level Authentication** (**NLA**) from inside the remote settings for Windows.

Summary

My hope with this chapter is to provide a concise, comprehensive walk-through of the steps that any DirectAccess administrator should take on any DirectAccess server. Completing these steps will serve as an excellent baseline for making sure that your server complies with best practices for DirectAccess, and following this guide will minimize your chances for running into unexpected errors or messages when you get further along into the DirectAccess configuration. Our next chapter should also be read in combination with this one to fully understand the scope of best practices available within DirectAccess, but this first chapter is definitely the hands-on input that you need to get started configuring it for yourself!

2
DirectAccess Environmental Best Practices

Now that we have the applicable knowledge of bringing a DirectAccess server online successfully and in a standardized fashion, let's talk about some other factors that you will definitely want to consider for your infrastructure. In this chapter, we will discuss some of the common points of confusion surrounding DirectAccess and its requirements, and I'll do my best to lay out the pros and cons to each aspect as I see them. There's no bias here, as you can see, I will plainly state that some of the "features" that are now available in 2012 are bad ideas in my opinion, and I'll recommend that you steer clear of some things to stick with a best practices approach.

We are going to talk about the following concepts:

- To NAT or not to NAT?
- Planning for Certificates (Public Key Infrastructure)
- Defining your **Group Policy Objects** (**GPOs**) and security groups
- Setting up the **Network Location Server** (**NLS**)
- Do I need IPv6 or ISATAP?
- Teredo and 6to4 tips and tricks

To NAT or not to NAT?

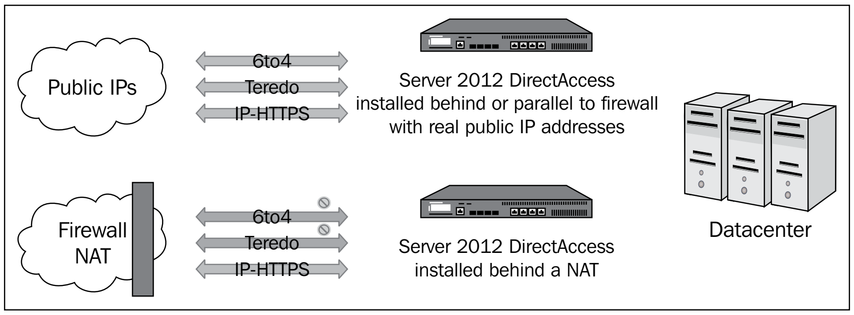
I give demonstrations and help companies plan the implementation of DirectAccess almost every day, and this question seems to be one of the hardest to answer for everyone, so let's tackle it first. Many of you who are working with Server 2012 DirectAccess are coming with some experience of the previous iteration, **Unified Access Gateway** (**UAG**) DirectAccess.

In running DirectAccess through UAG, there was a hard requirement for the server to have two public IP addresses on the external network interface. These had to be true public, Internet IPv4 addresses, and they had to be consecutive. I have personally never had a customer here in the US who had any trouble coming up with the necessary addresses, but I have read that this was an issue for some folks out there, and so it makes sense that Microsoft would try to address this "blocker to implementation" and allow the DirectAccess server to be placed behind a NAT.

Before we talk in detail on this, I'd like to provide a little extra insight into the UAG requirements for this, so that you understand what exactly is affected. The requirement for two consecutive public addresses is a firm requirement in the UAG software interface; it will not let you proceed with the DirectAccess configuration without meeting that requirement, but it's not actually a requirement for DirectAccess at all. The two IPs are a requirement for the Teredo protocol, one of the three ways that DirectAccess client computers can connect to the server to establish their tunnels. Teredo uses echo requests and bounces data off both IP addresses to be able to determine what kind of NAT the user is currently sitting behind, and therefore which configuration it needs to utilize to connect their Teredo tunnel to the DirectAccess server successfully.

In Server 2012 DirectAccess, you no longer need to provide two consecutive public IP addresses on the external NIC. In fact, you don't have to specify a public IP address at all! You can have the external interface configured with a private address (in a DMZ, or wherever), and you can use a NAT on your existing firewall to shoot the packets from the outside through to this interface. This obviously helps out those companies who are running short on public IP addresses, as it allows you to utilize DirectAccess without needing to find more. In some cases in the past, it was also a challenge for the security team to accept a UAG server being placed right out on the Internet, in parallel with the perimeter firewall, so that it could have these public IP addresses, even though UAG includes a very robust firewall on its own. These security teams will no doubt see the ability to place the Server 2012 DirectAccess server behind a NAT as a security benefit as well.

So far it's all good, right? Now, let's talk about the downsides. I made this diagram, so that during an online webinar or demo, I could help folks quickly visualize the downside to implementing your DirectAccess server behind a NAT.


Public IPs
6to4
Teredo
IP-HTTPS
Server 2012 DirectAccess installed behind or parallel to firewall with real public IP addresses
Datacenter
Firewall NAT
6to4
Teredo
IP-HTTPS
Server 2012 DirectAccess installed behind a NAT

As you can see, placing your external interface behind a NAT results in only one out of the three transition technologies being available for your DirectAccess clients to use to establish their connectivity. While IP-HTTPS is a great technology and connects even in situations where 6to4 and Teredo may not otherwise connect, I see the implementation of a DirectAccess server behind a NAT to be less than desirable for two reasons.

Three is better than one

Putting yourself in a position where IP-HTTPS is the only available technology by which client computers can establish their DirectAccess connections will work great, until it doesn't. Configurations get changed, certificates expire, things happen, and eventually you may find yourself in the position where a variable in the IP-HTTPS configuration has changed for some reason, and it stops working. If your DirectAccess server is configured with two public IP addresses so that you have the use of the other transition technologies, you may not even notice this partial failure of your environment for a while, because in my experience, most DirectAccess connections out there in the wild make use of the Teredo protocol for connectivity by default. There are a number of different things that can happen to take down the IP-HTTPS listener. Most common is the expiry of the SSL certificate that is used. I have received this phone call far too often where DirectAccess has stopped working for a company, and it turns out that the SSL certificate that is in use on the server for the IP-HTTPS listener expired, because nobody had it in their calendars to check on that guy. What is most interesting about these calls is that most of the time, even in large organizations; it is typically at least a few days (a few months in one case!) after the expiry date before they notice this happening. That is because they had Teredo available to them, and so the majority of their remote computers continued to connect without any problem at all, only those computers that were sitting in networks where Teredo was being blocked, and so they were forced to rely on IP-HTTPS were experiencing the problems.

On the other hand, I probably don't even need to say it at this point, because it's pretty obvious, in a DirectAccess implementation where you have deployed behind a NAT, and therefore IP-HTTPS is the only protocol available for your remote clients to connect; when IP-HTTPS goes down, everybody goes down. DirectAccess stops working for everybody, right now. I know a lot of IT guys that own motorcycles, presumably because they don't experience a lot of excitement sitting behind a desk. I can tell you that the rush of adrenaline that you get from that phone call and frantically trying to fix IP-HTTPS for thousands of users at one time is just as much of a rush. So there you go; thrill seekers among you will absolutely want to implement behind a NAT, and at some point, you'll thank me.

Efficiency of Teredo over IP-HTTPS

Each of the three transition technologies that DirectAccess uses takes a slightly different approach for preparing the packets for sending across the Internet. Because of these differences, each protocol is slightly different in terms of efficiency and speed. We'll give just a brief description of each here for some context.

6to4

The 6to4 tunneling protocol is used when the DirectAccess client computer has a real public IP address on the Internet. This doesn't happen very often these days, for the most part the only places I see public IPv4 addresses being issued directly to client computers are cell cards plugged into laptops, or occasionally hotels that offer the special **Click this button for a VPN capable internet connection** option on their web-based authentication page. In those rare cases, that 6to4 is chosen as the connectivity platform for DirectAccess; it takes the IPv6 packets and encapsulates them inside IPv4 using Protocol 41 before sending them across the Internet.

Teredo

Teredo is similar to 6to4. It encapsulates IPv6 packets into IPv4 headers, so that they can make their way across the IPv4 Internet, but Teredo does so using UDP; port 3544 to be precise. Teredo is able to connect when the user is sitting behind a NAT, such as at home, or in a coffee shop, and so it is used much more often by the DirectAccess computers than 6to4 is. In fact, on any given DirectAccess server which is configured to allow all three transition technologies, there is a very good chance that at any time of any day, the majority of your connected computers will be doing so using the Teredo transition technology.

IP-HTTPS

If there is a reason that neither 6to4 nor Teredo can connect, then it's the job of IP-HTTPS to pick up the slack. A good example of this is a wireless router at a coffee shop that blocks everything except for web traffic. The client will have a private IP address, which means 6to4 won't work, and if the router is blocking UDP, then Teredo won't work either. IP-HTTPS works by taking the IPv6 packets, encapsulating them in IPv4 headers, then encapsulating them again in HTTP headers, and encrypting them using TLS/SSL, essentially making the packet stream HTTPS. In this way, it goes out over TCP 443, just like any other HTTPS traffic, and can make its way successfully across those routers running higher security thresholds. So, you can see that IP-HTTPS is doing more grunt work to get those packets ready for transmission. Thus, it is a less efficient protocol. Furthermore, HTTPS traffic is obviously encrypted with SSL, but DirectAccess packets are already IPsec encrypted. So the packets are being encrypted twice, which makes it far less efficient, enough so that the client will notice a speed difference between Teredo and IP-HTTPS.

Now, before you get all huffy with me, yes, I do know about the new feature in Server 2012 DirectAccess that allows the second encryption to be `null`. The previous paragraph about IP-HTTPS performing double encryption has always been very relevant to anyone running a Windows 7 / UAG DirectAccess environment, because it will result in a slower connection for the users and will put much more stress on the UAG server to handle all of that encryption and decryption processing. One of the additions to the DirectAccess technology in Server 2012 is that the SSL part of the encryption processing can now be `null` encryption, which means that the double encryption no longer happens! DirectAccess packets are still always IPsec encrypted, so you are still completely protected, but the speed with which IP-HTTPS runs is now almost on par with Teredo, except for the one extremely special note that is not very well known: The IP-HTTPS null encryption performance enhancement in Server 2012 only benefits Windows 8 client computers. Your Windows 7 clients, and in my experience, so far, everybody still has Windows 7 clients, will still do the double encryption and will still result in IP-HTTPS being a slower connectivity method, which still places more load onto your DirectAccess server. This is the second reason that I recommend going with the public IP implementation whenever possible and giving yourself Teredo as a connectivity option for your users.

Planning for Certificates (PKI)

Certificate planning is traditionally the most confusing aspect of DirectAccess for anybody coming into it fresh, and unfortunately, I have spoken to dozens of administrators who tried to create a lab or proof-of-concept for DirectAccess and gave up, because they couldn't figure out what was actually necessary in their **Public Key Infrastructure** (**PKI**), or the requirements looked too complicated to be worth their time and the project was abandoned. Don't worry! If you have a Windows **Certificate Authority** (**CA**) server in your environment, you meet the requirements. And even if you don't, it is very easy to setup a CA server. I think the worst part about certificates with DirectAccess is figuring out which information is the right information, and knowing what you actually do (and don't) need. Microsoft also recognized this as a problematic area, and so the Server 2012 DirectAccess wizards do allow you to take an extremely simple path to implementation that doesn't require any certificate work to be done at all, but I definitely don't recommend that; we are going for the best practices approach after all! In a typical DA environment, there are only three certificates that you need to be concerned with, and two of those three are likely certificates that you are already familiar with handling.

SSL certificate for NLS

The first certificate that we need isn't for the DirectAccess server at all, but rather this is just a standard SSL certificate like you would put on any other website, and it gets installed onto the server where you are hosting your **Network Location Server** (**NLS**). We'll actually talk a little more shortly about why the NLS website should be hosted externally to your DirectAccess server, but for the purposes of this section, you just need to know that you will have a website in your network, it'll be running on a webserver (most likely IIS), and it must be an HTTPS site, so it requires a valid SSL certificate. The Subject Name of this certificate must match whatever DNS name you chose for the NLS website, and the intended purpose of this certificate must be Server Authentication. Again, this is similar to any other SSL certificate. In most implementations, we issue this certificate from your internal CA server. This is really just to minimize costs. The only computers that will be contacting this website will be your DirectAccess client computers that are inside the network, so there is no need to externally publish your PKI or **Certificate Revocation Lists** (**CRL**), and this certificate can be simply issued from the internal CA server without any other considerations. I have seen a few customers place a certificate that was purchased from a public authority on their NLS website, particularly places where they already own a wildcard certificate that can be used as many times and places as they choose, so that is definitely also an option if you are more comfortable doing it that way.

The DirectAccess wizards do allow you to utilize a self-signed certificate for this task, and they even allow for the hosting of your NLS website on the DirectAccess server itself. Please do not choose either of these options, as we are aiming for a best practices and most secure implementation of DirectAccess, and neither of those options fit the bill.

SSL certificate for IP-HTTPS

This is another simple requirement. As we already mentioned, the traffic from the client computers that connect via IP-HTTPS will essentially be HTTPS traffic, and so just as any webserver, the DirectAccess server requires an SSL certificate installed onto it to validate those connections. This is a standard SSL certificate that you can import into either the Certificates MMC or directly into IIS, and the Subject Name of this certificate must match whatever name you are going to enter into the DirectAccess wizards as the public DNS name for the connection. Or it can definitely be a wildcard, if you have one available. There is one statement that I always like to make regarding the IP-HTTPS certificate that will save you from a lot of headaches. **Use an SSL certificate for IP-HTTPS that was purchased from a public authority**. Unlike the NLS certificate, this guy is going to be validated by client computers connecting over the Internet, and they do like to verify connectivity to the CRL. Because of this, if you choose to try and utilize an IP-HTTPS certificate that was issued from your internal PKI, you must externalize some resources so that your CRLs are publically accessible, or your DirectAccess connections will not work. I have seen countless people try and fail to utilize a certificate from an internal CA server for this purpose, and so I absolutely recommend that you pony up the relatively small cost for this certificate and purchase it from GoDaddy, Entrust, VeriSign, or wherever you normally source your certificates.

Yes, the DirectAccess wizards even allow you to utilize a self-signed certificate for the purposes of IP-HTTPS authorization. Please don't do this in a production environment!

Machine certificates for IPsec

Now, we are down to the final and the most complicated piece of the PKI puzzle, as it relates to DirectAccess anyway. This is the part that can easily steer you off course during your planning, so I'll try to cut out the non-critical information give it to you straight. **You need a Windows CA server**. And…that's it. We will, of course, talk some more about what you need to do with that CA server, but I like to stress that when you throw all of the PKI/certificate documentation that is out there into a big ole pot and boil it down, the single core requirement that you want to make sure you meet is simply the existence of a Windows server in your domain that has the Certification Authority role installed. If you don't have one, it takes five minutes to make one. In many larger networks we would probably find a complex PKI topology with offline roots and intermediate CAs and backups to the backups, but none of that is necessary, and DirectAccess doesn't really care one lick about it. We are going to use your internal CA server to issue a machine certificate out to the DirectAccess server or servers, and to each of your DirectAccess client computers that are going to be connecting in from outside. This machine certificate is going to be used to validate the IPsec tunnels, those tunnels cannot be built without the existence and successful validation of this machine certificate.

Wait a minute; I can hear some of you yelling at me again. Yes, I do know there is a new feature in Windows 8 and Server 2012, called **Kerberos Proxy** (**KerbProxy**) that has the potential to negate the need for these machine certificates in particular networks. This feature is really for very basic implementations, and in my experience is mostly useful for a lab or proof of concept, but isn't a reality in most production networks. I'll explain this with a list. If you ever want to accomplish any of the following items in your DirectAccess environment, then you have no choice. You must distribute and utilize the machine certificates in your DirectAccess environment, if any of the following are true:

- You want to connect Windows 7 computers via DirectAccess (See what I did there? 98% of you just decided that you need machine certificates)
- Multisite DirectAccess is appealing to you
- You need to load balance a DirectAccess server cluster
- Two-factor authentication is in your future
- Force Tunneling sounds like fun

You get the idea. If you want to send Windows 7 clients through DirectAccess, or do any of the advanced features, you must use machine certificate authentication. It also goes without saying that using certificates is simply more secure than not using certificates.

Requirements for the machine certificate

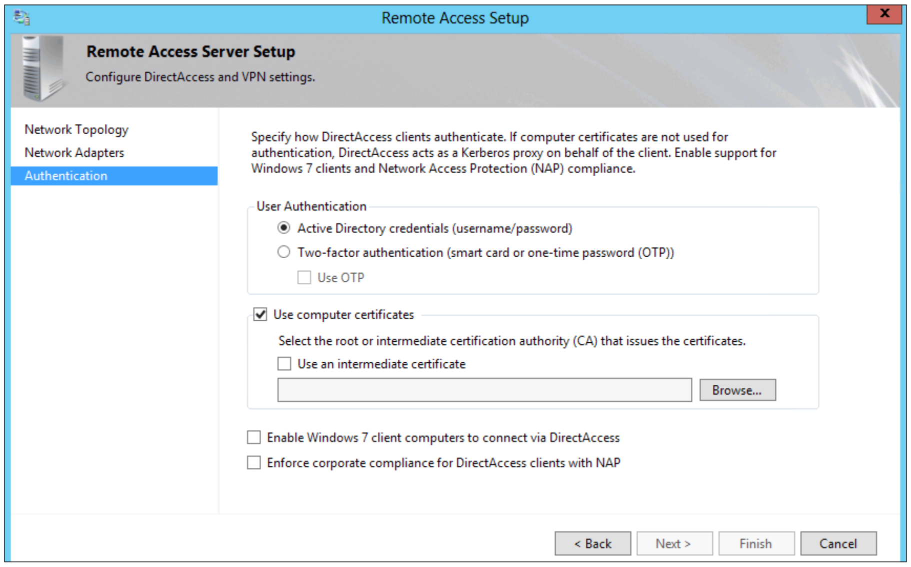
When you request a certificate from your CA server, what you are really doing is choosing a **Certificate Template** from a list of templates that exist on that CA server, and based on the settings that are configured inside that template; a certificate is then built and issued down to your server or computer. Installing the CA server role on a Windows server automatically includes a number of predefined certificate templates. So whether you have an established PKI infrastructure, or just finished installing that role onto a new server, chances are you already have made available a Template that is named **Computer**. This predefined template, which you did not have to do anything to configure, does exactly what we need it to do for validating DirectAccess tunnels. So the majority of DirectAccess implementations out there are making use of this template, and that works great. In this case, all you have to do is issue certificates to all the machines based on this template, and you are done. However, you may have some desire not to utilize that template, in which case you can certainly head over to the CA server and create your own template. In creating your own template to validate DirectAccess connections, there are four criteria that I like to point out, which you will want to meet:

- It must serve the Intended Purpose of Client Authentication
- It must also serve the Intended Purpose of Server Authentication
- The Subject Name of the certificate must be the **Common Name** (**CN**) of the machine (the FQDN of the computer to which it is being assigned)
- The **Subject Alternative Name** (**SAN**) of the certificate must be the DNS Name of the machine (also the FQDN of the computer to which it is being assigned)

Choosing the CA in the wizards

In *Chapter 1*, *DirectAccess Server Best Practices*, when we were talking about all of the specific steps that you want to take on your server, we detailed the step-by-step process for getting this machine certificate onto the server, so we certainly won't cover that again. This section has been the environmental background on PKI interacting with DirectAccess, more of the *why* rather than the *how*. But there is one server-specific screenshot of which I want to make note, and I do this because this particular screen in the DA configuration wizards always causes confusion. This is the **Authentication** screen, the last section of step 2 in the DirectAccess configuration wizards, where you choose the option to require machine certificates as part of the tunnel authentication process. The reason this screen is confusing is because everyone assumes that we will be presented with a list of certificates, and we choose the certificate that was issued by the internal CA server.

Unfortunately, that would be too easy. What you actually need to choose from this list is the name of the CA server that issued the machine certificate. Keep in mind that this doesn't necessarily reflect the DNS name of that server, but rather the actual name of the CA that is listed in Active Directory. The following is the screen where you are going to choose the option to utilize certificates for authentication:



Specify how DirectAccess clients authenticate. If computer certificates are not used for authentication, DirectAccess acts as a Kerberos proxy on behalf of the client. Enable support for Windows 7 clients and Network Access Protection (NAP) compliance.

User Authentication

- [✓ Active Directory credentials (username/password)]
- Two-factor authentication (smart card or one-time password (OTP))
- Use OTP

[✓] Use computer certificates

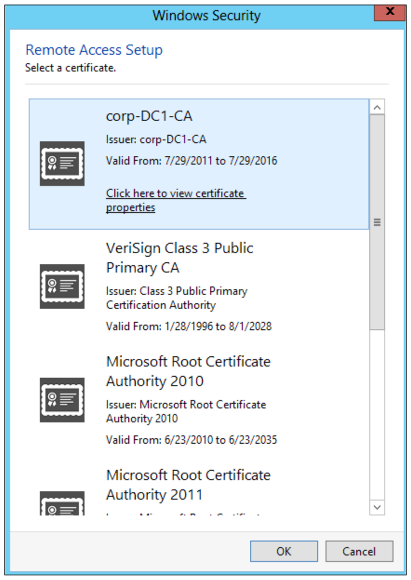
Select the root or intermediate certification authority (CA) that issues the certificates.

Use an intermediate certificate

Enable Windows 7 client computers to connect via DirectAccess

Enforce corporate compliance for DirectAccess clients with NAP

The following is a sample list of what you may see when clicking on **Browse...**.



corp-DC1-CA
Issuer: corp-DC1-CA
Valid From: 7/29/2011 to 7/29/2016
Click here to view certificate properties

VeriSign Class 3 Public Primary CA
Issuer: Class 3 Public Primary Certification Authority
Valid From: 1/28/1996 to 8/1/2028

Microsoft Root Certificate Authority 2010
Issuer: Microsoft Root Certificate Authority 2010
Valid From: 6/23/2010 to 6/23/2035

Microsoft Root Certificate Authority 2011

If you aren't sure about what to select in the wizards, particularly in large PKI environments, you can double-click on the certificate itself and navigate to the **Certification Path** tab, where the certificate chain should clue you in as to what CA issued your certificates.

Marking your calendars for certificate expirations

Making sure that your certificates don't expire sounds like common sense, but you would be amazed by the number of phone calls I receive each year from customers whose DirectAccess has unexpectedly stopped working, and we track it down to an expired certificate of some kind. The machine certificates are not likely to cause you any grief, as they are generally configured by default to auto-renew whenever their expiration is upcoming. This is really an issue with the two SSL certificates. If either the NLS certificate or the IP-HTTPS certificate expires, it has the potential to stop DirectAccess from working, and maybe worse it has the potential to stop those laptops from being able to communicate while the users are inside the office. Consider that a preview, as we will talk more extensively about that issue in our unique troubleshooting cases later in the book.

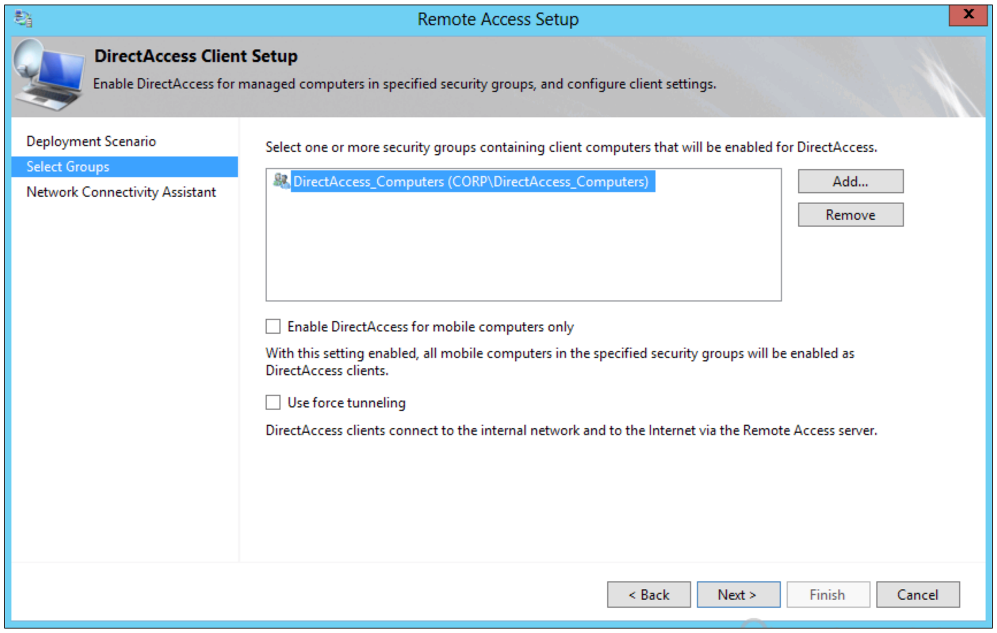
Defining your GPOs and security groups

I talk to a lot of people about DirectAccess. One of the most fun questions to field is "How do you install the client software". You probably know by now that there is no client software necessary, but I love it when I inform them of this and their response is "but there must be some kind of client software". Cracks me up every time. No, this is not a VPN, where you have to go through the grunt work of installing, configuring, and updating software when new releases come out, and fixing software when it breaks. I'm so sorry to all of you VPN admins out there who have to deal with this every day, because it doesn't have to be this way! Since DirectAccess clients and servers must be domain joined, it makes sense that they can take advantage of features in Active Directory, features such as Group Policy for the automatic distribution of configuration settings out to the machines.

If you are running Windows 7 Ultimate or Enterprise, or Windows 8 Enterprise, you already have the technical components to make DirectAccess work baked right into your operating system! Your "client software" is already installed; all you have to do is tweak it to get it to work! The greatest part? You don't even have to touch those laptops to turn them into DirectAccess clients. One of the very last things that you do when running through the DirectAccess configuration wizards on the server is defining a couple of Group Policy Objects. These GPOs will contain all of the connectivity settings that are needed to get DirectAccess up and running for both the clients and for the DA servers. You have a couple of different options available to you for creating these GPOs.

Let the wizards take care of it

If you follow the default settings in the wizards regarding these policies, the DirectAccess configuration tool will take care of everything for you. Upon clicking **Finish**, the very last step in the wizard, scripts will be run that create the GPOs, populate the GPOs with settings, and even set security filtering on those GPOs, so that they are only applied to the machines that are involved with the DirectAccess environment. For example, you obviously don't want externalized remote access connectivity settings being applied to your domain controllers. The configuration wizard will know which group or groups to security filter the settings to based on the **Select Groups** screen inside Step 1, where you defined which Active Directory group or groups would contain your DirectAccess client computers.



DirectAccess Client Setup

Enable DirectAccess for managed computers in specified security groups, and configure client settings.

Select one or more security groups containing client computers that will be enabled for DirectAccess.

DirectAccess_Computers (CORP\DirectAccess_Computers)

☐ Enable DirectAccess for mobile computers only

With this setting enabled, all mobile computers in the specified security groups will be enabled as DirectAccess clients.

☐ Use force tunneling

DirectAccess clients connect to the internal network and to the Internet via the Remote Access server.

For example, if you have a group called `DirectAccess_Computers` which you specified in Step 1, that group would be injected into the **Security Filtering** section of the DirectAccess Client GPO, so that those settings were only applied to machines which you add to that group.

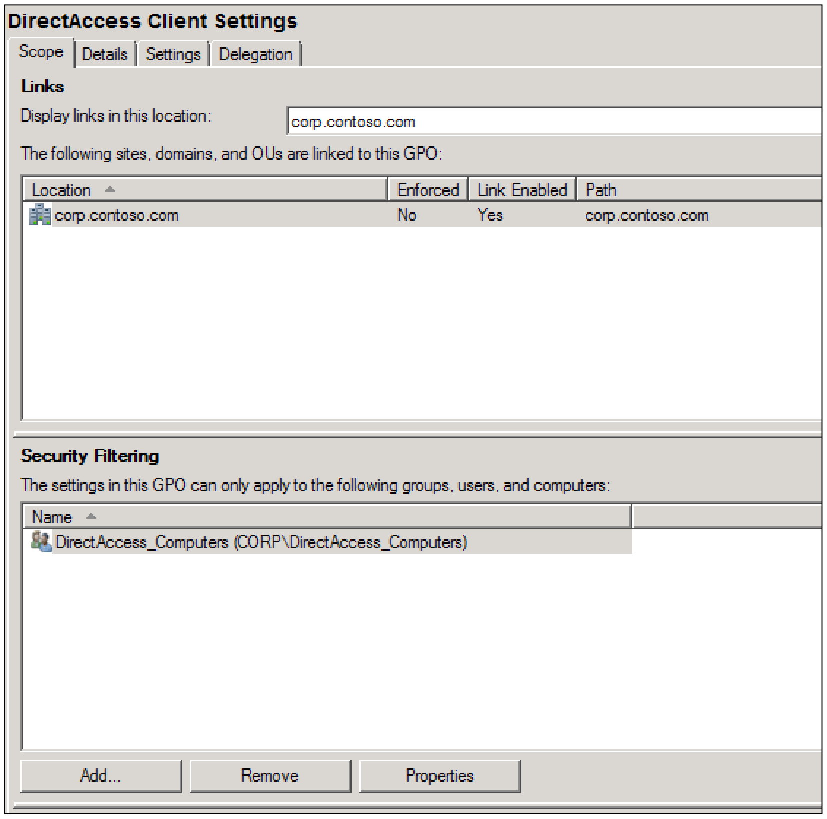
> Once again, don't use the Getting Started Wizard! The GPO filtering is not accommodated, for in a very sensible way, when implementing DirectAccess via the shortcut method. Use the real wizards, define your own filtering settings, and you'll be happy you did.

Creating your own GPOs

Allowing the Remote Access Management wizards to create and manage the GPOs for you works very well, and is obviously very appealing because they are completely hands-off, and you don't have to worry about administering those objects. So why is there even a reason to use your own GPOs? Sometimes, you don't have a choice, based on your company's written policies. I have installed in places where the DirectAccess admin working on the server doesn't have any rights or control in Active Directory, and in those cases we have to request that GPOs be created for us by the appropriate team in the company. So that is a legitimate reason you may have to take this approach. Other than that, one common reason that I see in the field for admins to prefer creating their own GPOs is so that you can link these GPOs to wherever you would like in the domain. You see, allowing the management console to manage the GPOs for you is very nice and easy, but one thing that it does automatically is link those GPOs at the root of your domain. This of course does not mean that those GPOs are going to apply to everything in your domain, because remember we have **Security Filtering** configured, so these settings are still going to apply to those machines only, which you have specified. Rather, the common complaint that I hear back from companies with reference to these GPOs being linked at the top of the domain is simply that their AD/Group Policy person doesn't like seeing them up at the top when opening Group Policy Management Console. To most of the admins, this makes no difference and seems a trivial argument, but some like to keep it looking clean, and perhaps to some level of standard on where the links do and do not exist, and regardless of the security filtering they would rather not see those links at the root of the domain. So whatever the reason, occasionally we want to make use of self-created GPOs for use with DirectAccess. This comes with some downsides, namely that you are now responsible for creating, linking, and filtering the GPOs yourself, by hand. Now that you have shunned the wizard from this task, you're on your own for these jobs. You also increase the potential for human error to make a disaster of your plans if those GPOs are not created properly, and could end up with some complex troubleshooting procedures to track down the problem.

To utilize your own GPOs, the first thing you want to do is to create two GPOs and link them to OUs that make sense to you. One GPO will contain the **DirectAccess Client Settings**, and the other GPO will contain the **DirectAccess Server Settings**. Name them whatever you would like, and create those links. Then make absolutely sure to set **Security Filtering** on these GPOs, so that the client settings only apply to the Active Directory group where you will be placing your DA client computers, and make sure that the server settings apply only to the DirectAccess server or servers. The following are some screenshots of the **Security Filtering** sections for both GPOs in a standardized environment to give you an idea of how they should look like.

The following is a typical configuration for the DirectAccess Client GPO's Security Filtering settings:


DirectAccess Client Settings

Links

Display links in this location: corp.contoso.com

The following sites, domains, and OUs are linked to this GPO:

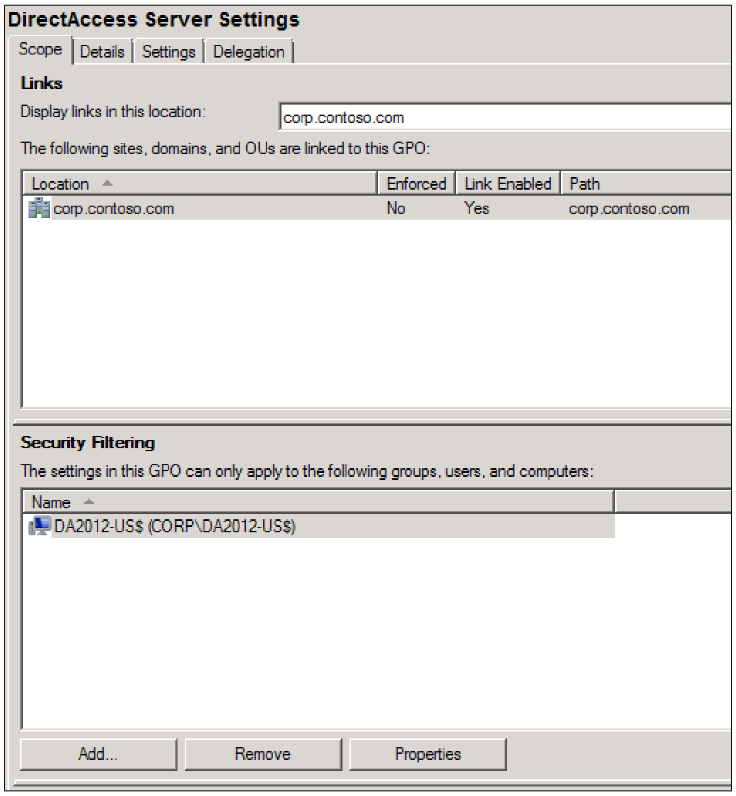
Location	Enforced	Link Enabled	Path
corp.contoso.com	No	Yes	corp.contoso.com

Security Filtering

The settings in this GPO can only apply to the following groups, users, and computers:

Name
DirectAccess_Computers (CORP\DirectAccess_Computers)

The following is a typical configuration for the DirectAccess Server GPO's Security Filtering settings:



DirectAccess Server Settings

Links

Display links in this location: corp.contoso.com

The following sites, domains, and OUs are linked to this GPO:

Location	Enforced	Link Enabled	Path
corp.contoso.com	No	Yes	corp.contoso.com

Security Filtering

The settings in this GPO can only apply to the following groups, users, and computers:

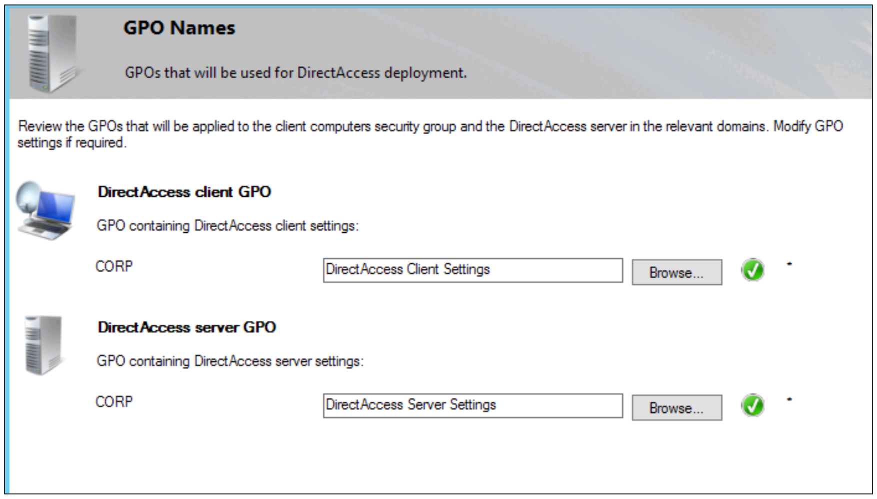
Name
DA2012-US$ (CORP\DA2012-US$)

Once your GPOs are ready for action, all we have to do is specify them inside the DirectAccess configuration wizards, so that the wizard populates them instead of creating its own GPOs. To do this, perform the following steps:

1. Run through the DirectAccess configuration steps 1 through 4, as with any DirectAccess implementation.
2. Once finished, the last step to make the configuration live is to press the **Finish** button on the bottom of the screen.
3. After clicking on **Finish**, you are presented with a summary screen of all the options that were chosen during the configuration wizards.

4. On this summary screen, you have the option to modify the GPO settings. Click on the **GPO Settings Change...** button near the top, and simply type in the exact names of your own GPOs that you want to utilize.

   The following is a screenshot of that screen:



GPO Names

GPOs that will be used for DirectAccess deployment.

Review the GPOs that will be applied to the client computers security group and the DirectAccess server in the relevant domains. Modify GPO settings if required.

DirectAccess client GPO

GPO containing DirectAccess client settings:

CORP: DirectAccess Client Settings

DirectAccess server GPO

GPO containing DirectAccess server settings:

CORP: DirectAccess Server Settings

The stars to the right of the naming fields will then indicate whether or not it has successfully identified and validated those GPOs. As long as this looks correct, you're good to go. Finish the wizard, let it do its thing, and your DirectAccess is live using those self-created objects.

Setting up the Network Location Server (NLS)

The Network Location Server is a very simple requirement in the DirectAccess environment, but a very critical one. NLS is just a website; it doesn't even have to be a dedicated server, that runs only inside the corporate network. It is not publicly accessible. Every time that the DirectAccess client computers get a network connection, they query this website. If they see it, they will know that they are inside the network, and that they do not need to turn on DirectAccess. Specifically, what this does is disables the **Name Resolution Policy Table** (**NRPT**), so that the name resolution requests do not attempt to be pushed over the DirectAccess tunnels, which wouldn't exist if you were inside the office. On the flip side, if your client computer cannot validate NLS, it assumes you are out in the wild and ready to fire up that DA connection in the background, and so that process is initiated automatically.

With Server 2012 DirectAccess, there is an option to host the NLS website right on the DirectAccess server itself. If there is one piece of advice that you take away from this section of writing, it is this. **Do not host the NLS website on the DirectAccess server**. This is another one of those options that is really intended for proof-of-concept installs, and not intended for production use. This is obvious even inside the wizards, where it will tell you that setting up the NLS website on a server other than the DA server is recommended. There are also certain implementation paths, such as network load balancing and multisite, where you don't have a choice; you must externalize NLS and put it on a different server. So just do it in the first place and make your job easier.

Standing up an NLS website is extremely easy. Choose a server and create a new website on it. Very commonly, we will track down an existing IIS server that is doing something on your network already, give it a new internal IP address to completely segregate the NLS traffic from other sites on that server, and turn on a new website on that IP address. The website can be anything you want it to be, typically I create a simple Default.htm file, inside which I type some simple text such as `This is the NLS server for DirectAccess, please do not delete me!`

I recommend *not* running the default IIS splash screen on the NLS site. I have personally witnessed a handful of sites, where NLS validation randomly failed for no apparent reason. In troubleshooting, it was discovered that by swapping out the IIS splash screen for a simple `Default.htm`, these problems disappeared.

Now, you can choose an internal DNS name for the site, maybe `nls.company.local`, and point this DNS record at the IP address where the site is running. At this point, you should be able to browse to `http://nls.company.local` from inside your network and see that text you typed, but we're not quite finished yet. You'll remember that we spent some time talking about the SSL certificate that needs to be placed on your NLS website. Yes, this is the same NLS website. So head into the site's properties in IIS, set up a 443 binding, and assign the appropriate SSL certificate to the site. Now, you should be able to browse to `https://nls.company.local`, and this is precisely what we need for successful NLS validation by the DirectAccess client computers.

Do I need IPv6 or ISATAP?

Part of the purpose of this chapter is to clear the air about requirements and give you definitive answers to some of the common questions that I receive all the time. "Do I need IPv6?" and "What's the deal with ISATAP?", are two of those questions, and the answers to those questions mesh and interweave. The direct answer is no. I know numerous companies running successful DirectAccess for thousands of users that have absolutely no IPv6 or ISATAP configured in their network. The only reason that you would want to introduce IPv6 or ISATAP inside your network is a situation where you need to initiate outbound communications from somewhere inside your network out to a DirectAccess-connected client computer. With the myriad of different Internet based solutions that exist for this kind of remote assistance or management, this is actually a surprisingly uncommon request. But there are certainly instances where "manage out" is desired or required, and if that is the case for you, there is quite a lot of information that I want to cover on the topic. The *Chapter 3*, *Configuring Manage Out to DirectAccess Clients*, will be dedicated to talking about this scenario in detail.

Teredo and 6to4 tips and tricks

You already know that when possible, you want users connecting through DirectAccess by utilizing the more efficient protocols. IP-HTTPS works very well, particularly for the Windows 8 clients, but let's discover a couple of quick tricks you can have up your sleeve to increase the utilization of Teredo as our preferred tunneling method.

Set Teredo to EnterpriseClient

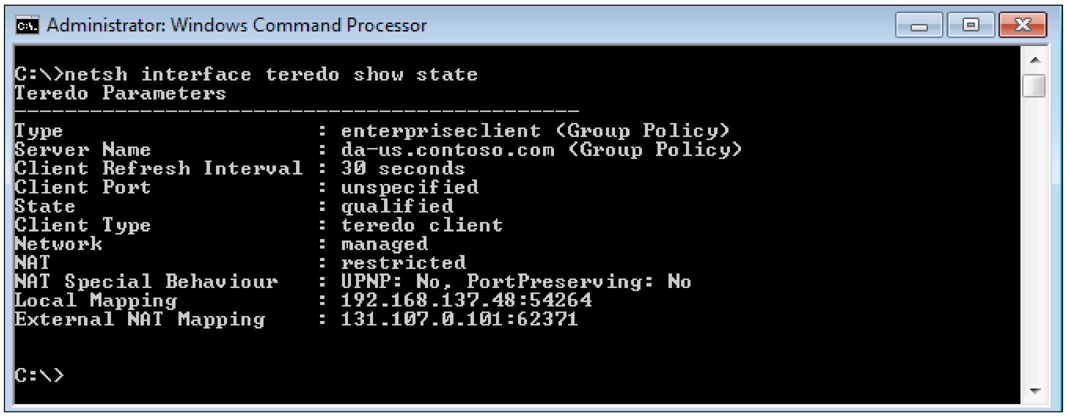
There are different modes, or statuses, that you can set Teredo to use on the DirectAccess client computers. By default, Teredo is set to `Client`. There is one big downside to running this way, if your DirectAccess client computer is currently plugged into a network that is a domain network (any domain, not just your own), Teredo will not connect by design. A good example of this is one of your users visiting a customer site and plugging in their laptop while they are there. If Teredo recognizes that a domain exists on that network, it will not connect. In this case, IP-HTTPS should pick up the slack and connect on its own accord, but what if there was a way to tell Teredo that it should connect even if it is sitting in a domain network? This is exactly the option that we are going to set here. By changing the status of the Teredo adapter on the client computers to EnterpriseClient, Teredo will then attempt to connect even while inside a domain network. If the customer is also blocking outbound UDP traffic, then that attempt will still fail and IP-HTTPS will still be used, but setting Teredo to EnterpriseClient does allow Teredo to be used in many more places than it is by default.

For testing purposes, or if you like administrative work, you can make this change on a per client basis by running the following simple `netsh` command:

```
netsh interface teredo set state enterpriseclient
```

This will immediately change Teredo from `Client` to EnterpriseClient on that particular computer. You can check the current status of the Teredo adapter anytime with another `netsh` command.

The following screenshot is an example output of the `netsh interface teredo show state` command:



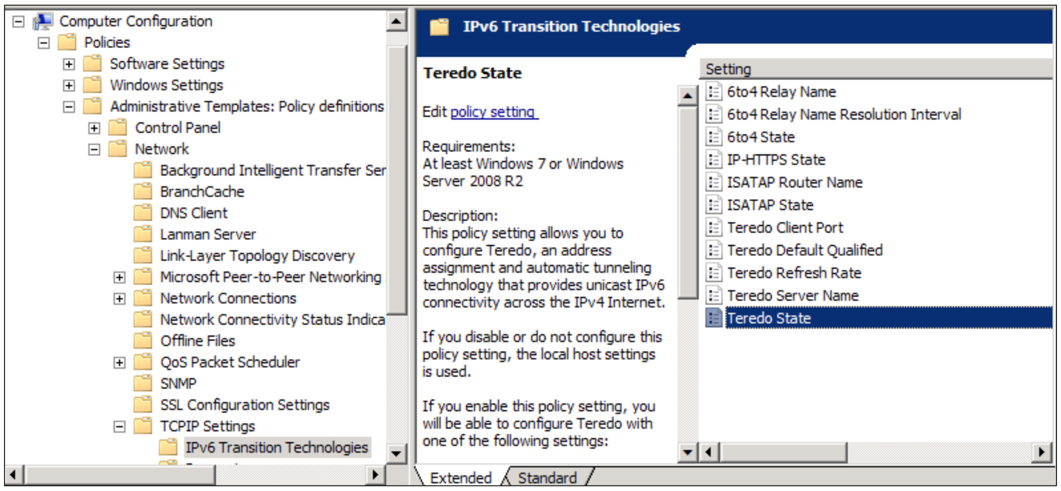
```
C:\>netsh interface teredo show state
Teredo Parameters
---------------------------------------------
Type                    : enterpriseclient (Group Policy)
Server Name             : da-us.contoso.com (Group Policy)
Client Refresh Interval : 30 seconds
Client Port             : unspecified
State                   : qualified
Client Type             : teredo client
Network                 : managed
NAT                     : restricted
NAT Special Behaviour   : UPNP: No, PortPreserving: No
Local Mapping           : 192.168.137.48:54264
External NAT Mapping    : 131.107.0.101:62371


C:\>
```

Using Group Policy for this change

Since Group Policy is awesome, we can use it to make this change on all of our DirectAccess client computers for us, so that we don't have to touch them. Simply create a new GPO, link and filter it so that it applies to your same group of computers to which the DirectAccess client settings are applying, and navigate to the following path:

Computer Configuration | Policies | Administrative Templates: Policy definitions | Network | TCPIP Settings | IPv6 TransitionTechnologies | Teredo State = Enterprise Client

The following is a screenshot of the placement inside the GPO, where you can configure the preceding setting:



IPv6 Transition Technologies

Teredo State

Edit policy setting

Requirements:
At least Windows 7 or Windows Server 2008 R2

Description:
This policy setting allows you to configure Teredo, an address assignment and automatic tunneling technology that provides unicast IPv6 connectivity across the IPv4 Internet.

If you disable or do not configure this policy setting, the local host settings is used.

If you enable this policy setting, you will be able to configure Teredo with one of the following settings:

I specified to create a new GPO purposely. Please *do not* modify the existing DirectAccess client settings GPO that is generated by DirectAccess. You should not be modifying this GPO by hand unless there is a specific reason that you must. It is much better and safer to apply the Teredo change from within its own policy.

Disabling the 6to4 adapter on your clients

This doesn't have any bearing on Teredo vs IP-HTTPS, but I have encountered dozens of cases over the years where DirectAccess tunnels fail to establish when the client is connected using the 6to4 protocol. Remember, 6to4 isn't used very often at all anyway, only when the client computer has a public IP address on its NIC, and so disabling it generally doesn't make any difference whatsoever in your environment. In my opinion, disabling it simply saves you a future headache, if you were to ever encounter this problem. The issue that I have experienced is when users are connecting to the Internet using cellular data. Typically, it's only with the USB cards or the ones built into the laptops, because those are the only ones that give out true public IP addresses on the Internet. The cell data systems that act more like wireless hotspots typically behave more like a normal router and hand private (NAT) IP addresses to the clients, and in that case 6to4 isn't going to be used anyway, and Teredo will do its thing. But, back to the issue at hand, if a DirectAccess client uses a cell card to connect their Internet and gets a public IP address, 6to4 will try to connect. Sometimes this works just fine.

Other times, the cell carrier will allow the initial handshake to happen, enough so that the client thinks that it has a successful 6to4 tunnel (so it decides to use it and not set up a Teredo tunnel), but then the carrier proceeds to block IP Protocol 41, which is what 6to4 uses to transmit packets. This results in a DA client being stuck on 6to4, because it thinks that the data can pass, but of course the user cannot open any resources. In this case, if we simply disable the 6to4 adapter on the laptop, it will immediately use Teredo instead, which works just fine on cell connections, and DirectAccess will jump into action.

Just like changing Teredo adapter settings, we can utilize `netsh` to adjust 6to4 on a PC-by-PC basis, as follows:

```
netsh interface 6to4 set state disabled
```

You can re-enable 6to4 just as easily if you are running some tests and want to do so, as follows:

```
netsh interface 6to4 set state enabled
```

Using Group Policy for this change

Once again, you can absolutely use a GPO to enable this change on all of your DirectAccess clients. In fact, you can certainly use the same GPO that you just created for setting Teredo to EnterpriseClient status, and just add this option in there as well. It is even located in the same place; navigate to the following path:

Computer Configuration | Policies | Administrative Templates | Network | TCPIP Settings | IPv6 Transition Technologies | 6to4 State = Disabled

Summary

I sincerely hope that this chapter was and continues to be a useful resource for making informed environmental decisions about the best possible way to implement DirectAccess in your environment. Perhaps, the most difficult thing about DirectAccess is the fact that there are so many different ways that it can be installed and used, and I am a firm believer in doing it *right*. Following these best practices will make your implementation as secure and performance oriented as possible. Now that DirectAccess is running smoothly in your network, let's get our hands a little bit dirtier with IPv6, as we discuss the steps and procedures that are required to get Manage Out functionality working. Next, we will work on getting resources inside your corporate network to be able to communicate outbound to your DirectAccess client computers that are roaming around in the wilderness.

3
Configuring Manage Out to DirectAccess Clients

DirectAccess is obviously a wonderful technology from the user's perspective. There is literally nothing that they have to do to connect to company resources; it just happens automatically whenever they have Internet access. What isn't talked about nearly as often is the fact that DirectAccess is possibly of even greater benefit to the IT department. Because DirectAccess is so seamless and automatic, your Group Policy settings, patches, scripts, and everything that you want to use to manage and manipulate those client machines is always able to run. You no longer have to wait for the user to launch a VPN or come into the office for their computer to be secured with the latest policies. You no longer have to worry about laptops being off the network for weeks at a time, and coming back into the network after having been connected to dozens of public hotspots while someone was on a vacation with it. While many of these management functions work right out of the box with a standard DirectAccess configuration, there are some functions that will need a couple of extra steps to get them working properly. That is our topic of discussion for this chapter.

We are going to cover the following topics:

- Pulls versus pushes
- What does Manage Out have to do with IPv6
- Creating a selective ISATAP environment
- Setting up client-side firewall rules
- RDP to a DirectAccess client
- No ISATAP with multisite DirectAccess

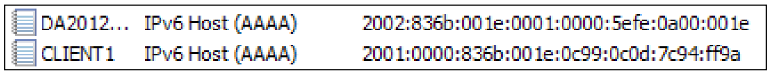
Pulls versus pushes

Often when thinking about management functions, we think of them as the software or settings that are being **pushed** out to the client computers. This is actually not true in many cases. A lot of management tools are initiated on the client side, and so their method of distributing these settings and patches are actually client **pulls**. A pull is a request that has been initiated by the client, and in this case, the server is simply responding to that request. In the DirectAccess world, this kind of request is handled very differently than an actual push, which would be any case where the internal server or resource is creating the initial outbound communication with the client, a true outbound initiation of packets. Pulls typically work just fine over DirectAccess right away. For example, Group Policy processing is initiated by the client. When a laptop decides that it's time for a Group Policy refresh, it reaches out to Active Directory and says "Hey AD, give me my latest stuff". The Domain Controllers then simply reply to that request, and the settings are pulled down successfully. This works all day, every day over DirectAccess. Pushes, on the other hand, require some special considerations. This scenario is what we commonly refer to as **DirectAccess Manage Out**, and this does not work by default in a stock DirectAccess implementation.

What does Manage Out have to do with IPv6?

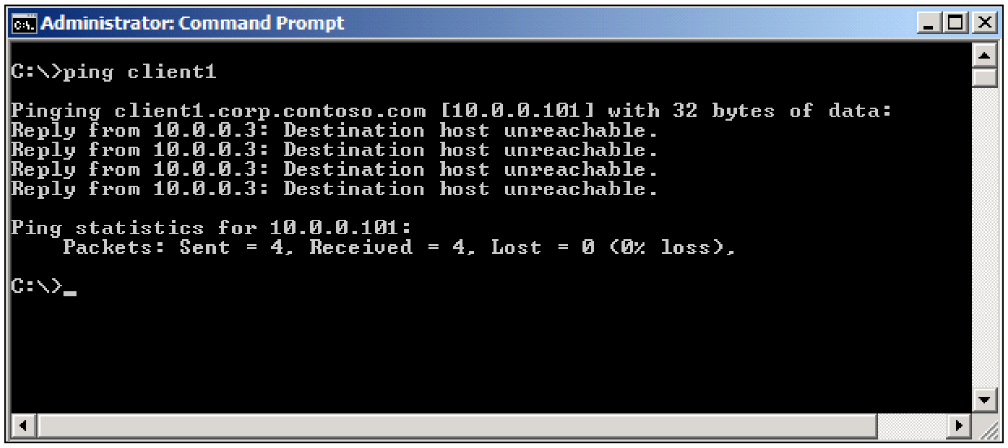
IPv6 is essentially the reason why Manage Out does not work until we make some additions to your network. "But DirectAccess in Server 2012 handles all of my IPv6 to IPv4 translations so that my internal network can be all IPv4, right?" The answer to that question is yes, but those translations only work in one direction. For example, in our previous Group Policy processing scenario, the client computer calls out for Active Directory, and those packets traverse the DirectAccess tunnels using IPv6. When the packets get to the DirectAccess server, it sees that the Domain Controller is IPv4, so it translates those IPv6 packets into IPv4 and sends them on their merry way. Domain Controller receives the said IPv4 packets, and responds in kind back to the DirectAccess server. Since there is now an active thread and translation running on the DirectAccess server for that communication, it knows that it has to take those IPv4 packets coming back (as a response) from the Domain Controller and spin them back into IPv6 before sending across the IPsec tunnels back to the client. This all works wonderfully and there is absolutely no configuration that you need to do to accomplish this behavior.

However, what if you wanted to send packets out to a DirectAccess client computer from inside the network? One of the best examples of this is a Helpdesk computer trying to RDP into a DirectAccess-connected computer. To accomplish this, the Helpdesk computer needs to have IPv6 routability to the DirectAccess client computer. Let's walk through the flow of packets to see why this is necessary. First of all, if you have been using DirectAccess for any duration of time, you might have realized by now that the client computers register themselves in DNS with their DirectAccess IP addresses when they connect. This is a normal behavior, but what may not look "normal" to you is that those records they are registering are AAAA (IPv6) records. Remember, all DirectAccess traffic across the internet is IPv6, using one of these three transition technologies to carry the packets: 6to4, Teredo, or IP-HTTPS. Therefore, when the clients connect, whatever transition tunnel is established has an IPv6 address on the adapter (you can see it inside `ipconfig /all` on the client), and those addresses will register themselves in DNS, assuming your DNS is configured to allow it.



DA2012...	IPv6 Host (AAAA)	2002:836b:001e:0001:0000:5efe:0a00:001e
CLIENT1	IPv6 Host (AAAA)	2001:0000:836b:001e:0c99:0c0d:7c94:ff9a

When the Helpdesk personnel types `CLIENT1` in their RDP client software and clicks on connect, it is going to reach out to DNS and ask, "What is the IP address of CLIENT1?" One of two things is going to happen. If that Helpdesk computer is connected to an IPv4-only network, it is obviously only capable of transmitting IPv4 packets, and DNS will hand them the DirectAccess client computer's A (IPv4) record, from the last time the client was connected inside the office. Routing will fail, of course, because CLIENT1 is no longer sitting on the physical network. The following screenshot is an example of pinging a DirectAccess connected client computer from an IPv4 network:



```
C:\>ping client1

Pinging client1.corp.contoso.com [10.0.0.101] with 32 bytes of data:
Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.

Ping statistics for 10.0.0.101:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

C:\>_
```

How to resolve this behavior? We need to give that Helpdesk computer some form of IPv6 connectivity on the network. If you have a real, native IPv6 network running already, you can simply tap into it. Each of your internal machines that need this outbound routing, as well as the DirectAccess server or servers, all need to be connected to this network for it to work. However, I find out in the field that almost nobody is running any kind of IPv6 on their internal networks, and they really aren't interested in starting now. This is where the **Intra-Site Automatic Tunnel Addressing Protocol**, more commonly referred to as **ISATAP**, comes into play. You can think of ISATAP as a virtual IPv6 cloud that runs on top of your existing IPv4 network. It enables computers inside your network, like that Helpdesk machine, to be able to establish an IPv6 connection with an ISATAP router. When this happens, that Helpdesk machine will get a new network adapter, visible via `ipconfig /all`, named ISATAP, and it will have an IPv6 address. Yes, this does mean that the Helpdesk computer, or any machine that needs outbound communications to the DirectAccess clients, has to be capable of talking IPv6, so this typically means that those machines must be Windows 7, Windows 8, Server 2008, or Server 2012. What if your switches and routers are not capable of IPv6? No problem. Similar to the way that 6to4, Teredo, and IP-HTTPS take IPv6 packets and wrap them inside IPv4 so they can make their way across the IPv4 internet, ISATAP also takes IPv6 packets and encapsulates them inside IPv4 before sending them across the physical network. This means that you can establish this ISATAP IPv6 network on top of your IPv4 network, without needing to make any infrastructure changes at all.

So, now I need to go purchase an ISATAP router to make this happen? No, this is the best part. Your DirectAccess server is already an ISATAP router; you simply need to point those internal machines at it.

Creating a selective ISATAP environment

All of the Windows operating systems over the past few years have ISATAP client functionality built right in. This has been the case since Vista, I believe, but I have yet to encounter anyone using Vista in a corporate environment, so for the sake of our discussion, we are generally talking about Windows 7, Windows 8, Server 2008, and Server 2012. For any of these operating systems, out of the box all you have to do is give it somewhere to resolve the name *ISATAP*, and it will go ahead and set itself up with a connection to that ISATAP router. So, if you wanted to immediately enable all of your internal machines that were ISATAP capable to suddenly be ISATAP connected, all you would have to do is create a single host record in DNS named *ISATAP* and point it at the internal IP address of your DirectAccess server. To get that to work properly, you would also have to tweak DNS so that *ISATAP* was no longer part of the global query block list, but I'm not even going to detail that process because my emphasis here is that you should not set up your environment this way.

Unfortunately, some of the step-by-step guides that are available on the web for setting up DirectAccess include this step. Even more unfortunately, if you have ever worked with UAG DirectAccess, you'll remember on the IP address configuration screen that the GUI actually told you to go ahead and set ISATAP up this way.

[Please do not create a DNS host record named ISATAP! If you have already done so, please consider this chapter to be a guide on your way out of danger.]

The primary reason why you should stay away from doing this is because Windows prefers IPv6 over IPv4. Once a computer is setup with connection to an ISATAP router, it receives an IPv6 address which registers itself in DNS, and from that point onward whenever two ISATAP machines communicate with each other, they are using IPv6 over the ISATAP tunnel. This is potentially problematic for a couple of reasons. First, all ISATAP traffic default routes through the ISATAP router for which it is configured, so your DirectAccess server is now essentially the default gateway for all of these internal computers. This can cause performance problems and even network flooding. The second reason is that because these packets are now IPv6, even though they are wrapped up inside IPv4, the tools you have inside the network that you might be using to monitor internal traffic are not going to be able to see this traffic, at least not in the same capacity as it would do for normal IPv4 traffic.

It is in your best interests that you do not follow this global approach for implementing ISATAP, and instead take the slightly longer road and create what I call a "Selective ISATAP environment", where you have complete control over which machines are connected to the ISATAP network, and which ones are not.

[Many DirectAccess installs don't require ISATAP at all. Remember, this is only used for those instances where you need true outbound reach to the DirectAccess clients. I recommend installing DirectAccess without ISATAP first, and test all of your management tools. If they work without ISATAP, great! If they don't, then you can create your selective ISATAP environment.]

To set ourselves up for success, we need to create a simple Active Directory security group, and a GPO. The combination of these things is going to allow us to decisively grant or deny access to the ISATAP routing features on the DirectAccess server.

Creating a security group and DNS record

First, let's create a new security group in Active Directory. This is just a normal group like any other, typically a global or universal, whichever you prefer. This group is going to contain the computer accounts of the internal computers to which we want to give that outbound reaching capability. So, typically the computer accounts that we will eventually add into this group are Helpdesk computers, SCCM servers, maybe a management "jump box" terminal server, that kind of thing. To keep things intuitive, let's name the group **DirectAccess – ISATAP computers** or something similar. We will also need a DNS host record created. For obvious reasons we don't want to call this ISATAP, but perhaps something such as **Contoso_ISATAP**, swapping your name in, of course. This is just a regular DNS A record, and you want to point it at the internal IPv4 address of the DirectAccess server.

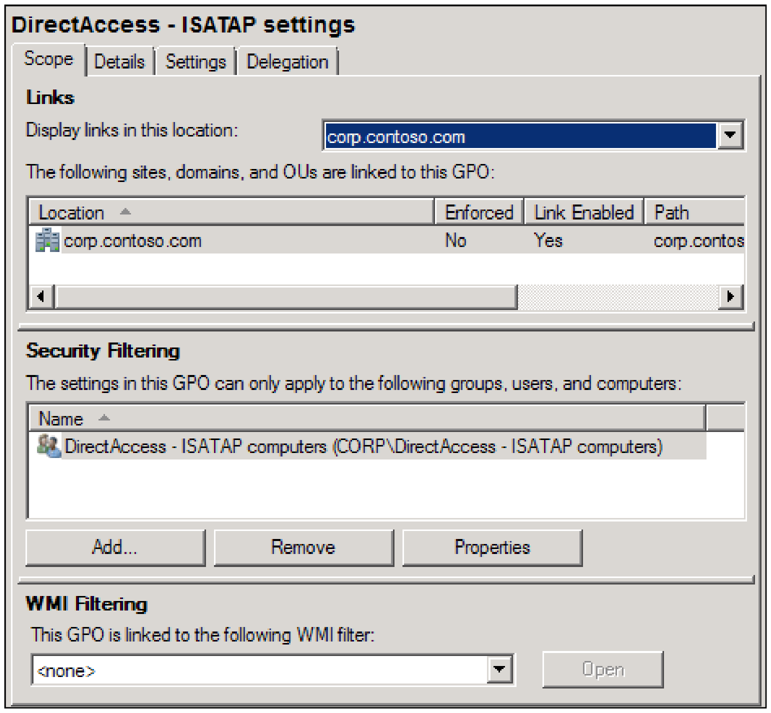
If you are running a clustered array of DirectAccess servers that are configured for load balancing, then you will need multiple DNS records. All of the records have the same name, Contoso_ISATAP, and you point them at each internal IP address being used by the cluster. So, one gets pointed at the internal **Virtual IP** (**VIP**), and one gets pointed at each of the internal **Dedicated IPs** (**DIP**). In a two-node cluster, you will have three DNS records for Contoso_ISATAP.

Creating the GPO

Now go ahead and follow these steps to create a new GPO that is going to contain the ISATAP connection settings:

1. Create a new GPO, name it something such as **DirectAccess – ISATAP settings**.
2. Place the GPO wherever it makes sense to you, keeping in mind that **these settings should only apply to the ISATAP group that we created**.
    - One way to manage the distribution of these settings is a strategically placed link.
    - Probably the better way to handle distribution of these settings is to reflect the same approach that the DirectAccess GPOs themselves take. This would mean linking this new GPO at a high level, like the top of the domain, and then using security filtering inside the GPO settings so that it only applies to the group that we just created. This way you ensure that in the end the only computers to receive these settings are the ones that you add to your ISATAP group.

I typically take the Security Filtering approach, because it closely reflects what DirectAccess itself does with GPO filtering. So, create and link the GPO at a high level, and then inside the GPO properties, go ahead and add the group (and only the group, remove everything else) to the **Security Filtering** section, like what is shown in the following screenshot:



DirectAccess - ISATAP settings

Links

Display links in this location: corp.contoso.com

The following sites, domains, and OUs are linked to this GPO:

Location	Enforced	Link Enabled	Path
corp.contoso.com	No	Yes	corp.contos

Security Filtering

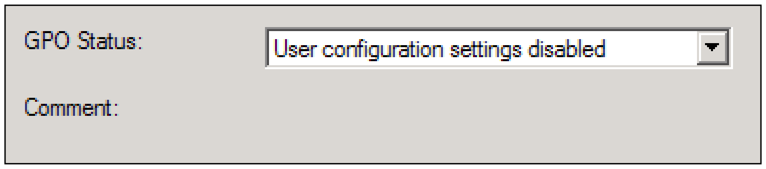
The settings in this GPO can only apply to the following groups, users, and computers:

Name
DirectAccess - ISATAP computers (CORP\DirectAccess - ISATAP computers)

WMI Filtering

This GPO is linked to the following WMI filter: <none>

Then move over to the **Details** tab and set the **GPO Status** to **User configuration settings disabled**.



GPO Status: User configuration settings disabled

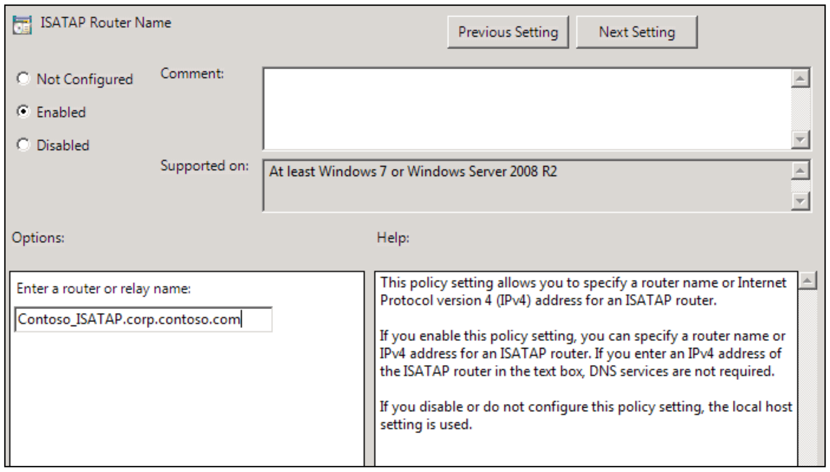
Comment:

Configuring the GPO

Now that we have a GPO which is being applied only to our special ISATAP group that we created, let's give it some settings to apply. What we are doing with this GPO is configuring those computers which we want to be ISATAP-connected with the ISATAP server address with which they need to communicate, which is the DNS name that we created for ISATAP.

First, edit the GPO and set the ISATAP Router Name by configuring the following setting:

Computer Configuration | **Policies** | **Administrative Templates** | **Network** | **TCPIP Settings** | **IPv6 Transition Technologies** | **ISATAP Router Name** = **Enabled** (and populate your DNS record).



[✓ Enabled]

Supported on: At least Windows 7 or Windows Server 2008 R2

Enter a router or relay name: Contoso_ISATAP.corp.contoso.com

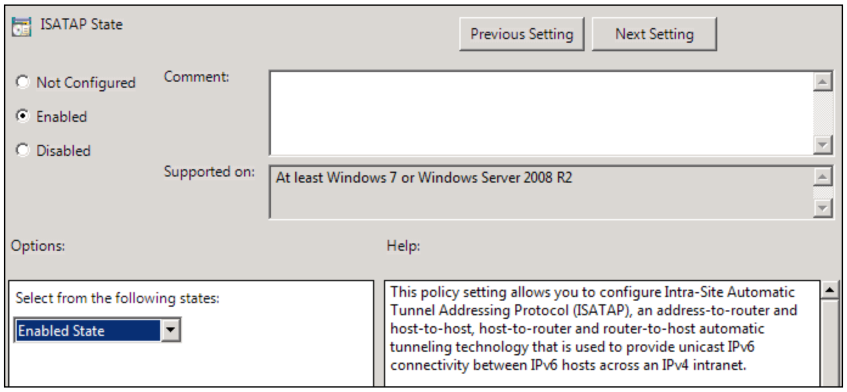
This policy setting allows you to specify a router name or Internet Protocol version 4 (IPv4) address for an ISATAP router.

If you enable this policy setting, you can specify a router name or IPv4 address for an ISATAP router. If you enter an IPv4 address of the ISATAP router in the text box, DNS services are not required.

If you disable or do not configure this policy setting, the local host setting is used.

Second, in the same location within the GPO, we want to enable your ISATAP state with the following configuration:

Computer Configuration | **Policies** | **Administrative Templates** | **Network** | **TCPIP Settings** | **IPv6 Transition Technologies** | **ISATAP State** = **Enabled State**.



Not Configured
[✓ Enabled]
Disabled

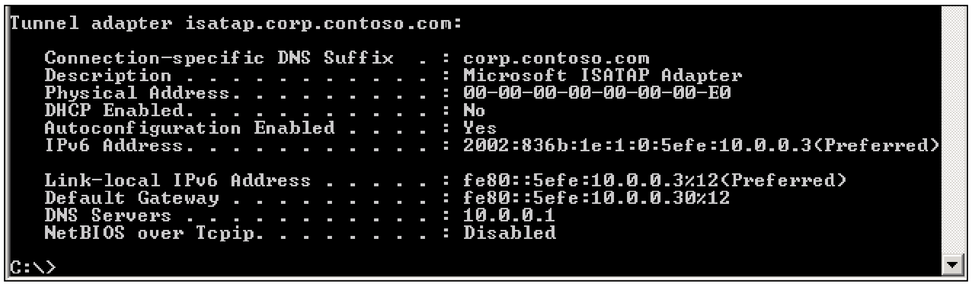
Supported on: At least Windows 7 or Windows Server 2008 R2

Select from the following states: Enabled State

This policy setting allows you to configure Intra-Site Automatic Tunnel Addressing Protocol (ISATAP), an address-to-router and host-to-host, host-to-router and router-to-host automatic tunneling technology that is used to provide unicast IPv6 connectivity between IPv6 hosts across an IPv4 intranet.

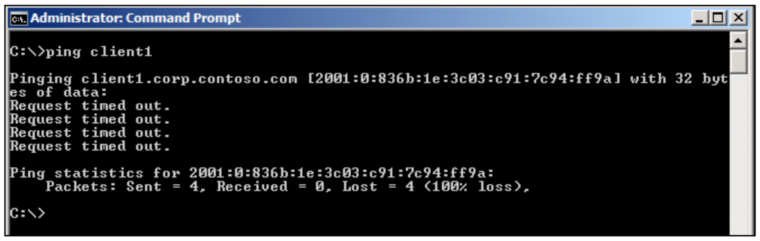
Adding machines to the group

All of our settings are now squared away, time to test! Start by taking a single computer account of a computer inside your network from which you want to be able to reach out to DirectAccess client computers, and add the computer account to the group that we created. Perhaps pause for a few minutes to ensure Active Directory has a chance to replicate, and then simply restart that computer. The next time it boots, it'll grab those settings from the GPO, and reach out to the DirectAccess server acting as the ISATAP router, and have an ISATAP IPv6 connection. If you generate an `ipconfig /all` on that internal machine, you will see the ISATAP adapter and address now listed as shown in the following screenshot:



```
Tunnel adapter isatap.corp.contoso.com:

   Connection-specific DNS Suffix  . : corp.contoso.com
   Description . . . . . . . . . . . : Microsoft ISATAP Adapter
   Physical Address. . . . . . . . . : 00-00-00-00-00-00-00-E0
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . : 2002:836b:1e:1:0:5efe:10.0.0.3(Preferred)

   Link-local IPv6 Address . . . . . : fe80::5efe:10.0.0.3%12(Preferred)
   Default Gateway . . . . . . . . . : fe80::5efe:10.0.0.30%12
   DNS Servers . . . . . . . . . . . : 10.0.0.1
   NetBIOS over Tcpip. . . . . . . . : Disabled

C:\>
```

And now if you try to ping the name of a client computer that is connected via DirectAccess, you will see that it now resolves to the IPv6 record. Perfect! But wait, why are my pings timing out? Let's explore that next.



```
C:\>ping client1

Pinging client1.corp.contoso.com [2001:0:836b:1e:3c03:c91:7c94:ff9a] with 32 byt
es of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 2001:0:836b:1e:3c03:c91:7c94:ff9a:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>
```

Setting up client-side firewall rules

Your internal ISATAP machine now has the ability to route packets out to the DirectAccess client computers through the ISATAP tunnel, but why on earth would the Windows Firewall that is running on those DirectAccess clients allow ICMP, RDP, SMB, or any traffic from this weird, IPv6-based ISATAP client that is all of a sudden trying to hit it? Our next and final step is to configure Windows Firewall with Advanced Security (WFAS) rules on the DirectAccess client computers so that they allow these communications from the internal ISATAP machines, instead of dropping those packets, like they do by default. I said it once, and I'll say it again, Group Policy is awesome, so let's use another GPO to define these WFAS rules.

[

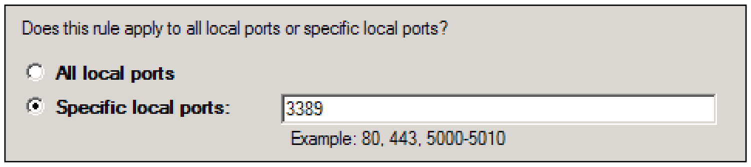
It is a common mistake to modify the existing DirectAccess Client Settings GPO that DirectAccess creates and uses, and to plug these new rules into that GPO rather than create another new GPO. Please don't do this. The DA GPOs should be left alone, because they are automatically adjusted by the wizards, so your changes may get thrown out at some point. Make sure to use a separate GPO for these WFAS settings. Perhaps the same one you created for the Teredo and 6to4 best practice settings, because these are also settings that need to be applied only to the DirectAccess client computers.

]

Inside the GPO that you have chosen for this task, this is the section where we can add some WFAS rules using following configuration:

Computer Configuration | **Policies** | **Windows Settings** | **Security Settings** | **Windows Firewall with Advanced Security** | **Windows Firewall with Advanced Security** | **Inbound Rules**

Just like you would do from inside the firewall console were you working straight from the client machine, you simply need to right-click and choose **New Rule…**. Most rules are going to be **Port** rules, then click on **Next**, then specify which port you would like to allow. You can either include multiple ports in one rule, or create multiple rules, one for each port.

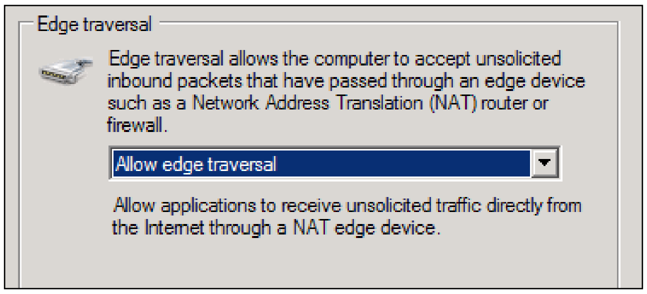


Does this rule apply to all local ports or specific local ports?

( ) All local ports
(•) Specific local ports: 3389
Example: 80, 443, 5000-5010

Choose **Allow the connection**, and on the screen where you define which Firewall Profiles to allow it to apply, you can choose all three if you would like, but ultimately we probably only care about applying these rules while the client is connected via DirectAccess. Anytime that a client computer is on DirectAccess, it will have either the **Public** or **Private** profile assigned, so selecting those two are the important ones. Finish out the wizard by naming the rule, and once your rule is complete, we need to right-click on the rule and head into its properties and change a couple of more advanced items.

On the **Advanced** tab, we need to change the **Edge traversal** drop-down menu to **Allow edge traversal**. This is important for making these rules function when the client is connected using the Teredo protocol.



Edge traversal

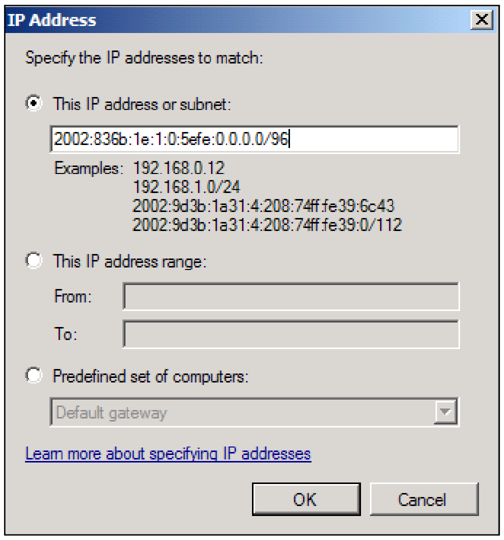
Edge traversal allows the computer to accept unsolicited inbound packets that have passed through an edge device such as a Network Address Translation (NAT) router or firewall.

Allow edge traversal

Allow applications to receive unsolicited traffic directly from the Internet through a NAT edge device.

Remember, you will have to create rules for each protocol that you want to be accessible when reaching out to these DA clients from the internal network. The most typical ones I see are RDP (TCP 3389), File Access (TCP 445), and ICMPv6, so that ping replies work.

At this point, the rule will work and you could move on to the next section where we test the RDP connection. However, the rule is currently allowing RDP from anywhere, and to tighten down security on that rule a little, we should really set it up so that this RDP rule is only allowing RDP connections from computers that are inside the ISATAP network. To do this, in the rule's Properties, we need to head into the **Scope** tab. Now, under the **Remote IP address** field, we want to change it to **These IP addresses**, and enter in the IPv6 prefix that our ISATAP environment uses. For me, the easiest way to determine that prefix is to look at the `ipconfig /all` that we did from the internal ISATAP connected computer (shown earlier in the chapter). In that screenshot, my ISATAP IPv6 address is **2002:836b:1e:1:0:5efe:10.0.0.3**—you notice that the end of this is my actual IPv4 address. For the prefix to add to this WFAS rule, we want to allow RDP connection from any ISATAP host, so our prefix is going to be `2002:836b:1e:1:0:5efe:0.0.0.0/96`.


IP Address

Specify the IP addresses to match:

[✓ This IP address or subnet:] 2002:836b:1e:1:0:5efe:0.0.0.0/96

Examples: 192.168.0.12
192.168.1.0/24
2002:9d3b:1a31:4:208:74ff:fe39:6c43
2002:9d3b:1a31:4:208:74ff:fe39:0/112

This IP address range:

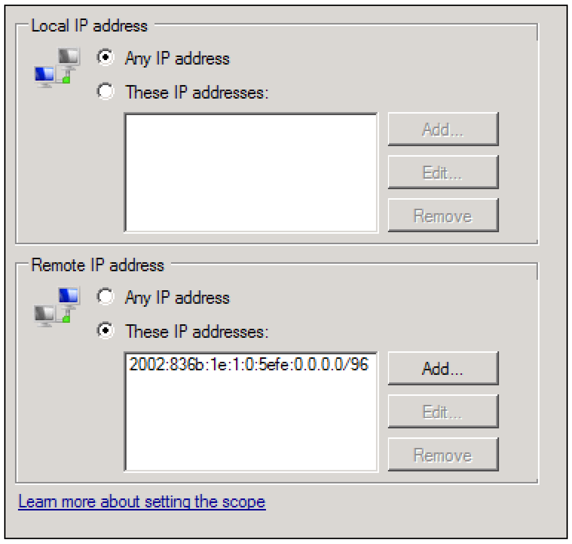
From:

To:

Predefined set of computers:

Default gateway

Click on **OK**, and you should see something similar to this in your **Scope**.



Local IP address

Any IP address [✓]

These IP addresses:

Remote IP address

Any IP address

These IP addresses: [✓]

```
2002:836b:1e:1:0:5efe:0.0.0.0/96
```

RDP to a DirectAccess client

IPv6 routing established via ISATAP tunnels? Check. Firewall rules on the DirectAccess clients to allow this traffic? Check. Now we are really ready to test. Log onto one of your newly furnished ISATAP machines inside the network, and try to contact a DirectAccess computer by name. A common first test is **ping**, which is fine, though if we get down into the nitty-gritty, we would find that ICMP traffic actually moves outside of the IPsec tunnels, and so a successful ping reply doesn't necessarily mean that everything is working. It usually does though. Otherwise, the most common testing platform is an RDP connection. This, of course, implies that you have enabled the system rule on those systems to allow RDP connections in the first place and that `3389` is one of the ports you allowed in the rules we created a few minutes ago. If those things are in place, you should now be able to open up the Remote Desktop Client from your ISATAP connected machine, type in the name of a DirectAccess client computer, have that name resolve to the IPv6 address the client is using via DirectAccess, and make a successful RDP connection to that box. Nice work!

No ISATAP with multisite DirectAccess

If you remember back a few pages, we talked about Windows preferring IPv6 over IPv4 whenever v6 is available. That, combined with the fact that an internal computer's ISATAP connection essentially default-routes all IPv6 traffic to its ISATAP router (the DirectAccess server) all the time, means that ISATAP and multisite DirectAccess don't really work together. If you had multiple DirectAccess entry points, each one of them running as an ISATAP router, your internal servers or PCs that you intend to join to the ISATAP network can only point one place. So they are going to hit either Site A or Site B, not both. This means that their responses to the client machines are always going to go out whichever site they are connected to for ISATAP. If a DirectAccess client is coming in through Site A and the Helpdesk computer is ISATAP connected to Site A, that will work fine. If a DirectAccess client is coming in through Site B, however, the packets from the Helpdesk computer will be trying to flow through Site A's DirectAccess server, which doesn't have IPsec tunnels out to the DA client, and that communication will fail. There used to exist some documentation from Microsoft on setting up ISATAP in a multisite fashion, where you offloaded the ISATAP router/server role onto its own equipment (its own Windows servers) in each site. As far as I know, this documentation has now been pulled because it was quite complicated to set up, and didn't really work as intended anyway. This is no longer a supported scenario, and I recommend you stay away from it. If you need outbound management access in a DirectAccess multisite environment, you will have to do real IPv6 inside your network and tap into that.

You can successfully use ISATAP in a cluster/array of DirectAccess servers that are on the same subnet. It is only when using multiple physical sites that you lose the ability.

Summary

ISATAP is typically misunderstood, and I hope this chapter has given you some straight-shooting information about its use in the real world. Many IPv6 guys consider ISATAP to be the red-headed stepchild of IPv6, but the reality is that there are many more companies today and in the future who are going to be running ISATAP environments, rather than real IPv6 environments. Once you know how to make ISATAP work for you and do what you want, it's really quite a useful tool. To anyone reading this that already has DirectAccess implemented in your network, check DNS for the existence of a record named ISATAP. If it exists, please take the time to delete it and set it up right, as per the instructions in this chapter. You may never thank me, because doing so means you never have to deal with strange routing or resolution problems down the road like you may have had to should you continue with your global approach.

4
General DirectAccess Troubleshooting

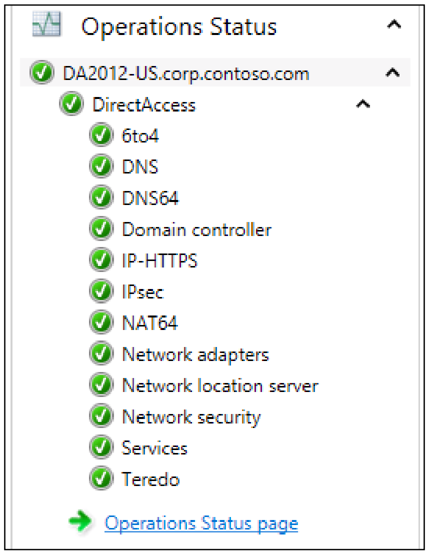
DirectAccess is typically a "set it and forget it" technology. Once you have all of the settings, certificates, and so on finished, and you have a successful DirectAccess client connected, there isn't a lot of day-to-day troubleshooting that you will encounter. Inevitably though you are bound to someday have a client that doesn't work for one reason or another, and you will want to have the capability to dig under the hood and figure out what is going on with that connection. Today, we are going to talk about the most common troubleshooting procedures you will use; including a close look over the extensive logfile that can be generated from the client computers should you ever have the need.

We are going to cover the following points:

- Remote Access Management Console
- Windows Firewall with Advanced Security
- Reading the client logfiles
- What happened to Teredo?
- Clients with native IPv6

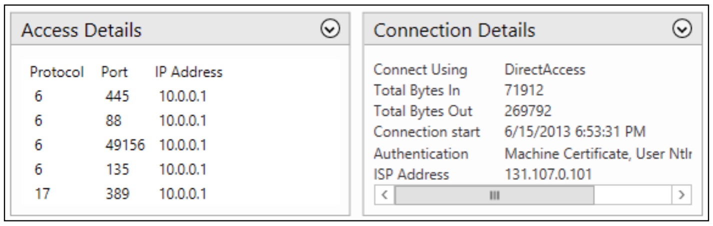
Remote Access Management Console

As common sense would dictate, there is plenty of information available right on the DirectAccess server that can help you diagnose and resolve problems you may encounter. Most of this information is available from inside the **Remote Access Management Console** page, which you can open from the **Tools** menu inside **Server Manager**. Once open, there are plenty of different tiers worth of data to examine. The dashboard gives you a quick overhead view of everything happening currently on the server. If you need to dig a little further, the **Operations Status** page is the most helpful for diagnosing server-side problems. This page employs nice "green check" versus "red x" status indicators for each separate technology that is being used by DirectAccess. When everything is running well, you should see something similar the following screenshot:



If anything is amiss with any one of these technologies, instead of the green checkmark you will see a red "x", and clicking it will give you some descriptive information about the issue.

The **Remote Access Management Console** page contains plenty of information, not only on the server side of the connection, but also grants you access to quite a bit of information about the client computers that are currently connected, or those that have been in the past. Currently connected client machines are visible inside the **Remote Client Status** screen. Here, you can see which computers are connected, what transition technology they are using to make their connection, who is logged into those computers, and even which internal resources they are accessing. If you click on one of the connected machines, look at the bottom of the console to see this information.



Access Details

Protocol	Port	IP Address
6	445	10.0.0.1
6	88	10.0.0.1
6	49156	10.0.0.1
6	135	10.0.0.1
17	389	10.0.0.1

Connection Details

Connect Using	DirectAccess
Total Bytes In	71912
Total Bytes Out	269792
Connection start	6/15/2013 6:53:31 PM
Authentication	Machine Certificate, User Ntlr
ISP Address	131.107.0.101

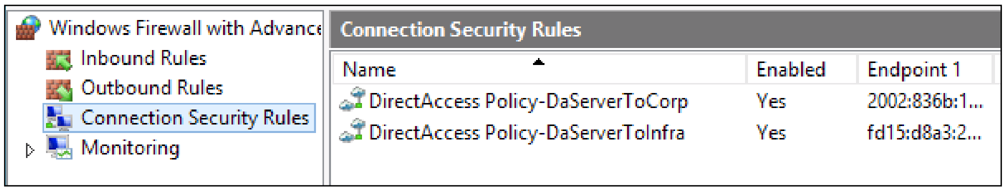
Unfortunately, you will have to do some reverse engineering of this information to figure out just what the users are up to. For example, it looks like my client is doing some LDAP, Kerberos, and file access to **10.0.0.1**, which happens to be my Domain Controller. So, it takes a little bit of work to make use of this information, but it's still very cool that it exists! The **Search** bar at the top of the screen is also very useful for filtering out specific users or computers for which you might be looking, or for example, to see all of the users who are currently connected using Teredo.

And then, the last screen available in the console is **Reporting**. This is not enabled by default, so make sure you go in there and turn it on. This function will give you the same information that you can see for active client connections, on a historical basis. So you can look back and generate reporting data to see who was connected over the past week, for example.

Windows Firewall with Advanced Security

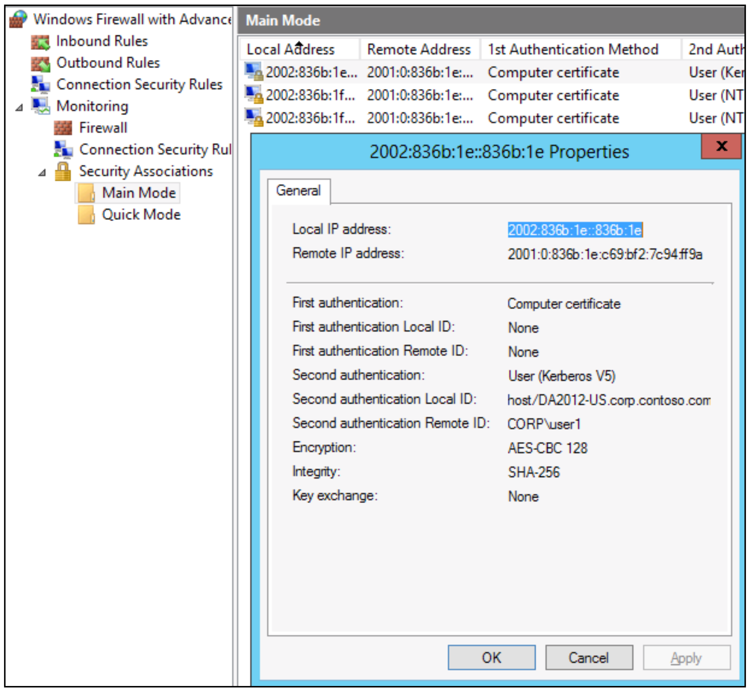
Another place on both the DirectAccess server and the clients that I frequent is **Windows Firewall with Advanced Security** (**WFAS**). From a command prompt, type `wf.msc`, and you will be presented with the WFAS console. Inside this console, the first screen we see is an "overhead view" of which firewall profiles are currently active. For the server, in our most common and recommended "edge" deployment, where you have an Internal and an External NIC, you should see both the **Domain** profile active (on the Internal NIC), and the **Public** profile active (on the External NIC). If you see any other behavior, you may have networking configuration issues or firewalls that are blocking your server's ability to contact the network resources that it needs. And then on the client side, it is important to know that the DirectAccess IPsec rules will only activate themselves when the NIC has been assigned either the **Public** or **Private** profiles, those rules will *not* be active if the laptop has the **Domain** profile running. So keep a look out for that as well. As long as profiles look good, you can head down into **Connection Security Rules**, and simply look for the existence of some rules here related to DirectAccess. These are the IPsec tunnel rules, and what we are checking here is to make sure that the settings from the GPOs got applied successfully. Whether you are looking on the server or the client, the IPsec rules that show up in this screen are sent down from inside the DirectAccess GPOs.

If you don't have rules here, you didn't get your GPO settings, and you'll need to figure out why.


Windows Firewall with Advance
Inbound Rules
Outbound Rules
Connection Security Rules
Monitoring
Connection Security Rules

Name	Enabled	Endpoint 1
DirectAccess Policy-DaServerToCorp	Yes	2002:836b:1...
DirectAccess Policy-DaServerToInfra	Yes	fd15:d8a3:2...

Now, expand the **Monitoring** folder and let's look at a couple of things here. First, you'll notice that there is yet another **Connection Security Rules** here. You can think of the **Monitoring** folder as a current active settings folder. Clicking on the **Connection Security Rules** option here will show you those same IPsec rules, if and when the **Public** or **Private** profiles are active. A working DirectAccess client or server will have DirectAccess IPsec rules listed here. Then, navigate to the **Security Associations** | **Main Mode** folder. This folder lists the active IPsec tunnels on this particular machine. If you are checking this on an externally connected client and there are no IPsec tunnels listed here, you have a problem that needs to be worked. On the server, you will be able to see everybody's IPsec tunnels in this folder. You can double-click on the tunnels to pull up additional information about those particular client machines and users.



Local Address	Remote Address	1st Authentication Method	2nd Auth
2002:836b:1e...	2001:0:836b:1e:...	Computer certificate	User (Ker
2002:836b:1f...	2001:0:836b:1e:...	Computer certificate	User (NT
2002:836b:1f...	2001:0:836b:1e:...	Computer certificate	User (NT

2002:836b:1e::836b:1e Properties

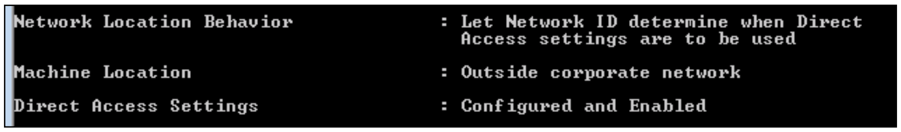
Local IP address:	2002:836b:1e::836b:1e
Remote IP address:	2001:0:836b:1e:c69:bf2:7c94:ff9a
First authentication:	Computer certificate
First authentication Local ID:	None
First authentication Remote ID:	None
Second authentication:	User (Kerberos V5)
Second authentication Local ID:	host/DA2012-US.corp.contoso.com
Second authentication Remote ID:	CORP\user1
Encryption:	AES-CBC 128
Integrity:	SHA-256
Key exchange:	None

Reading the client logfiles

Both Windows 7 and Windows 8 have tools which can generate some very helpful logfiles when troubleshooting a DirectAccess connection. In Windows 7, it is named the **DirectAccess Connectivity Assistant** (**DCA**), and is a systray program that you have to install. In Windows 8, it is named the **Network Connectivity Assistant** (**NCA**) and is now baked right into Windows, so you don't have to install anything. This NCA shows up in the list of Network Connections that appears when you click on your networking icon, the same place you would go in Windows 8 to view your available Wi-Fi connections. Whichever operating system you have and therefore whichever tool you are using, the logs generate a lot of the same information. Both logfiles are really just a collection of commands that are run, and the results stored, in one big file. You can certainly use these commands manually on a one-by-one basis if you are trying to gather some specific information, rather than look over a whole logfile. The following is some information regarding the most important commands that are included in the logfiles, which you can view from inside the logfiles, or run each command separately from a command prompt window:

- `Ipconfig /all`: Here's a great place to start, one with which everybody is already familiar. What you might not be as familiar with is all of the IPv6 information that you are going to see in a DirectAccess client's output. The important things to look for here are that the client is getting a real IP address in the first place, and then that the client is getting at least one transition adapter IPv6 address. So listed under either the 6to4, Teredo, or IP-HTTPS adapters, you should see at least one IPv6 address listed. Without this, nothing else with DirectAccess is going to work. You may see more than one IPv6 address listed, and that is fine. It is fairly common, for example, to see both a Teredo and IP-HTTPS connection active at the same time, at least temporarily while the system is booting.
- `Netsh interface teredo show state`: To see a little bit more information about Teredo, take a look here. This can be useful for figuring out whether or not the GPOs have been successfully applied (is **Server Name** populated? If so, that setting came from the GPO). You can also check **Type** here to determine whether or not you have set Teredo to the EnterpriseClient status yet, as we talked about in our **Best Practices** section. If Teredo is having trouble connecting, you'll get specific information about it here.

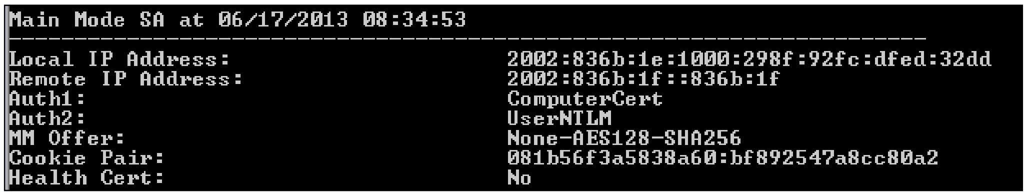
- `Netsh interface httpstunnel show interface`: This is the detailed information about the IP-HTTPS adapter on your computer. Useful for seeing the name of the IP-HTTPS listener running on your DirectAccess server, as well as whether or not the adapter is currently connected. If Teredo is connected, this guy probably won't be active and it'll tell you as much. If IP-HTTPS is having trouble connecting, it will list an error code that you can easily search on web to figure out why it's not connecting. Something that I very commonly do with the information in this command is, grab the URL that is listed in this command, and paste it into my web browser from an externally connected computer (DirectAccess or not). By doing this, I can query the IP-HTTPS interface and make sure it's responding. Although I won't get a web page presented by design, I will be able to make sure that I don't get a network timeout message which would indicate the traffic isn't flowing properly. And I can also use the browser to perform a double-check on my SSL certificate to make sure the name, date, and placement of the certificate are all in order.
- `Netsh dnsclient show state`: I use this command for two things. First, **Machine Location** will tell you whether the client thinks of itself as inside or outside the network, which can be very useful for troubleshooting. Second, **Direct Access Settings** will indicate for you whether or not the DirectAccess settings are configured and whether they are enabled or disabled. A normal externally connected DirectAccess client will look like the following screenshot:



```
Network Location Behavior            : Let Network ID determine when Direct
                                       Access settings are to be used

Machine Location                     : Outside corporate network

Direct Access Settings               : Configured and Enabled
```

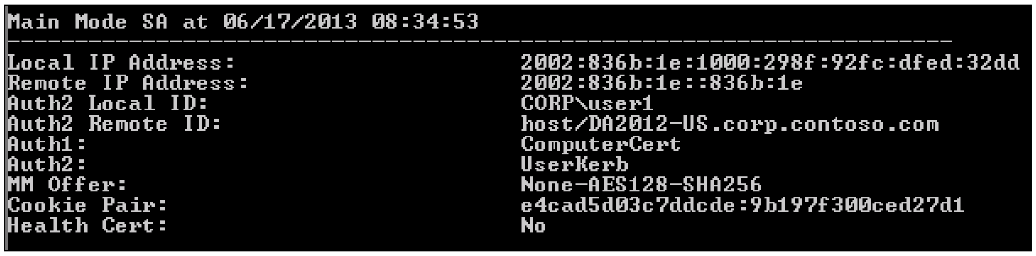
- `Netsh namespace show policy`: The output of this command should reflect the settings that you entered into the NRPT, during the **DNS** screen that was inside step 3 in the DirectAccess configuration wizards. Simply make sure that the information is populated here. If it is not, you likely have not received your settings from your GPO yet.
- `Netsh namespace show effectivepolicy`: When the DA client is outside of the network, this command should show you the exact same output as the `netsh name show policy` command, because the NRPT will be active and working. When the DA client is sitting inside the network, this command will say **Note: DirectAccess settings would be turned off when computer is inside corporate network**, indicating that the client is inside the network, and therefore does not have the NRPT enabled. If your client is sitting inside the network, yet shows all of the NRPT information in the output of this command, you are likely having name resolution problems at the moment, and the trouble most likely lies with your NLS server not working properly.

- `Netsh advfirewall monitor show mmsa`: This is just another view of the active IPsec tunnels on the system. We can also see this information from inside WFAS, but if you are sitting at the command prompt already, this is a quick, easy way to check for the existence of tunnels. Pay close attention to the **Auth2** fields in these IPsec tunnels. Infrastructure tunnels will look like the following screenshot:



```
Main Mode SA at 06/17/2013 08:34:53
----------------------------------------------------------------------
Local IP Address:                     2002:836b:1e:1000:298f:92fc:dfed:32dd
Remote IP Address:                    2002:836b:1f::836b:1f
Auth1:                                ComputerCert
Auth2:                                UserNTLM
MM Offer:                             None-AES128-SHA256
Cookie Pair:                          081b56f3a5838a60:bf892547a8cc80a2
Health Cert:                          No
```

The Intranet tunnels looks like the following screenshot:



```
Main Mode SA at 06/17/2013 08:34:53
----------------------------------------------------------------------
Local IP Address:                     2002:836b:1e:1000:298f:92fc:dfed:32dd
Remote IP Address:                    2002:836b:1e::836b:1e
Auth2 Local ID:                       CORP\user1
Auth2 Remote ID:                      host/DA2012-US.corp.contoso.com
Auth1:                                ComputerCert
Auth2:                                UserKerb
MM Offer:                             None-AES128-SHA256
Cookie Pair:                          e4cad5d03c7ddcde:9b197f300ced27d1
Health Cert:                          No
```

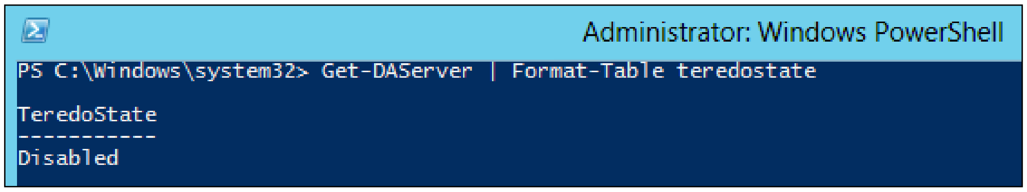
Sometimes, you may be trying to troubleshoot something, such as access to domain controllers working, but access to file servers fails, and you'll want to check in here and make sure that you have both kinds of tunnels. If you have only infrastructure tunnels listed, you could be having trouble acquiring your Kerberos ticket, perhaps because you are trying to contact a DC that is not available from the DirectAccess server or that is not part of the Management Servers list. When troubleshooting a connection, I often check the output of the IPsec tunnels, because if you see a successful Intranet tunnel listed here that is displaying the username of the person logged into the computer, then most likely DirectAccess is working just fine on their computer, and whatever trouble they are having lies beyond DA.

- `Netsh advfirewall show currentprofile`: I like this command, because it gives a quick glance into what **Firewall** profile is currently active on the client machine. As you know, the IPsec tunnels will only enable themselves if the Public or Private profiles are active, so make sure this is the case. If you are currently being assigned the **Domain** profile and are wondering why DA isn't working, there's your answer. Also, check this page to make sure the **State** field says **ON**. There are lots of companies who have the Windows Firewall disabled by existing Group Policy settings, and the firewall being set to **OFF** will absolutely mess up the DirectAccess connection.
- `Certutil -store my`: Here, we list all of the certificates that are stored on the local computer. What we want to make sure here is that our machine certificate exists and is valid. You should be able to see a certificate that was issued from your internal CA server, which has a Subject Name matching your client's FQDN, and ensure that said certificate has a valid date.
- `Systeminfo`: Just about the only thing I have ever looked at in `Systeminfo` is **OS Name**. Use this to make sure that the client's operating system is one that actually supports DirectAccess. You would be amazed by how many logfiles I have reviewed that showed Windows 7 Pro or Windows 8 Pro – this isn't going to work!
- Anyone familiar with the logfiles knows by now that I am listing, in order, most of the commands that come from DCA. You might find this interesting and backwards, because DCA is the older utility, but I like the logfiles better. NCA includes a few extra commands and outputs that are useful in certain situations, but for getting down to the root of why DA isn't working on a client, the DCA logs and specifically the above commands are all I need 99 percent of the time. Once you are familiar with these commands, you will start to catch very quickly when the output of one of them doesn't look right. After a while, you will be able to take any DA computer that isn't connecting, ask the user to email you a logfile, and be able to determine from that logfile alone what exactly is not working with the connection.

What happened to Teredo?

Now that a lot of folks are moving to Server 2012 DirectAccess, I find environments all the time where all of the client computers are using IP-HTTPS for their transition tunneling technology. If Teredo is the most commonly used protocol out there in the wild, why does this happen? It is usually because a "shortcut" method was taken to implementing DirectAccess in the first place. If you have installed your DirectAccess server with only a single public IP address, or no public IP addresses at all and stuck it behind a NAT, then there is your answer. In either of these deployment scenarios, your only transition technology available to use is IP-HTTPS. Also, if you have checked the **Force Tunnel** checkbox during the wizards to push all traffic through the DA tunnels, this mode also locks you down to only IP-HTTPS. But, I also find many DA servers out there that do have the two consecutive public IP addresses, which should cause Teredo to work, but all of the clients are IP-HTTPS. The most likely reason this is happening is that pesky Getting Started Wizard. When you use the GSW to implement DirectAccess, it is kind enough to decide for you that Teredo is unnecessary, and doesn't turn it on. So you're stuck with only IP-HTTPS clients connecting. Thankfully, you don't have to wipe your configuration and start over to correct this behavior (though you can take that approach as well); there is a simple command you can run to enable it after the fact. First, let's launch PowerShell on your DirectAccess server and make sure that Teredo really is set into the **Disabled** state by navigating to the following path:

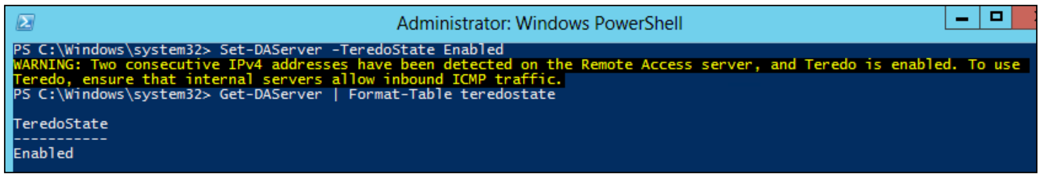
Get-DAServer | Format-Table teredostate



```
PS C:\Windows\system32> Get-DAServer | Format-Table teredostate

TeredoState
-----------
Disabled
```

Now that we know the Getting Started Wizard did indeed set Teredo to disabled by default, another simple PowerShell command, as follows, can enable it and correct that behavior:

```
Set-DAServer -TeredoState Enabled
```



```
PS C:\Windows\system32> Set-DAServer -TeredoState Enabled
WARNING: Two consecutive IPv4 addresses have been detected on the Remote Access server, and Teredo is enabled. To use
Teredo, ensure that internal servers allow inbound ICMP traffic.
PS C:\Windows\system32> Get-DAServer | Format-Table teredostate

TeredoState
-----------
Enabled
```

Now, your clients who are in networks that allow Teredo will start to set themselves up with Teredo adapter connectivity instead of IP-HTTPS, which will increase their performance (particularly for Windows 7 clients), and decrease load on the DirectAccess server.

Clients with native IPv6

The past year of so has shown an increasing amount of DirectAccess client computers that are receiving a native IPv6 address for internet browsing, for one reason or another. What usually happens is that I receive a call about a particular client computer that isn't working at a customer, and I ask that they ship me some logfiles. In those logfiles, right away in the `ipconfig`, I can see that there is a native IPv6 address listed on one of the regular network adapters. Typically, these addresses start with `2600:`, and they can show up on either a LAN adapter or wireless adapter, if this address came from a home router, for example. Otherwise they can also show up on the adapter installed from a cell phone card. I would say the cell cards are more common from what I have seen, but the core of the problem is the same in either case. If you see this IPv6 address, and you will probably also see a `2001:` or `2002:` address listed for either Teredo or IP-HTTPS as you normally would; head down to the section of the logs where you can see the IPsec tunnels. Many times, I have seen it where the IPsec tunnels are trying to establish using the `2600:` address, instead of the `2001:` or `2002:` as they are supposed to do. Because that native IPv6 address doesn't have any kind of connection to your DirectAccess server, those tunnels obviously fail. Why do the DA tunnels seem to prefer the native IPv6 address over the transition technology IPv6 addresses? I have no idea, and nobody has been able to give me a reason for it. The resolution is to remove the native IPv6 address from the client computer. There are multiple ways you can do this. You can disable IPv6 DHCP on the home router if that is the guy issuing it, or you can uncheck **TCP/IPv6** from inside the NIC properties on the computer. This also works for most of the cell cards, as they show up as a physical NIC inside **Network Connections** on the computer. You can open **Properties** of it and simply uncheck **TCP/IPv6**, and the card will stop issuing an IPv6 address to the client, and only use IPv4 plus the transition technologies. In all cases, as soon as we stop this native IPv6 address from being applied to the client computer, DirectAccess immediately starts working.

Summary

This chapter is dedicated to common troubleshooting procedures and situations that come directly from field work. DirectAccess has many moving parts, and a hiccup in any of those parts can cause symptoms or trouble for the users, and you will want to be aware of the places you can visit to discover the core problem causing these symptoms, and be empowered to fix them. I hope this was informative, and that it can be a quick reference guide for anyone in a DirectAccess administrator position. Now, it's time to move away from the standard, cookie-cutter methods of troubleshooting and dig into some of the unique situations I have found myself in over the years working with DirectAccess.

5
Unique DirectAccess Troubleshooting Scenarios

This chapter is going to reflect some of the unique and perhaps strange scenarios that I have encountered in the field when working with DirectAccess. Some of these cross my desk fairly regularly, and some are definitely from left field. Either way, they are all potential cases that you may someday encounter, and during our discussion of these specific topics you will also gain some more in-depth knowledge about the way that DirectAccess works, how name resolution is handled, and the flow of packets in the tunnels. Knowledge is power! Let's get some.

What we are going to talk about:

- What happens when NLS is offline?
- I enabled NLB and DA broke!
- IPv4 applications don't connect over DA
- Cannot contact some servers

What happens when NLS is offline?

Let's start with one of the craziest situations that can happen in a DirectAccess environment. This one is particularly nutty because when it happens, the symptoms that you experience are for your computers INSIDE the office, while your remote workforce continues to function normally. The NLS is the mechanism by which all of your DirectAccess client computers validate when they are inside the network. It is a very simple requirement (just a website), but if anything goes wrong with the validation of that website, crazy stuff happens. Any DirectAccess client computer that is sitting inside the office will not realize that it is inside the office, and will continue to leave the NRPT enabled, which results in all corporate DNS requests attempting to resolve themselves to the DirectAccess server's external interface, which isn't routable because the user is inside the network.

> There have been rare occasions where you can actually connect to DA from inside your own network, but it is the exception. If this is the case in your network, then if NLS goes down, your DirectAccess clients will still work when they are inside the office, but traffic will be flowing in the IPsec tunnels through the DirectAccess server instead of flowing natively over your LAN.

External DA clients will continue to function normally, because they can't see NLS by design when they are outside, and so while you are troubleshooting these strange name resolution problems happening on your internal network, the DirectAccess server is probably going to be the farthest thing from your mind. Unfortunately, it is normally hours after the problem starts that I receive the call that the customer figured out the commonality that it is only the DA clients having this trouble, and ask me to look into it. It typically takes about 5 seconds to diagnose over the phone as soon as I hear the symptoms, and it of course makes sense to them as soon as they hear it, but I point this out only because it can cause an enormous headache if you don't know for what to look.

The resolution

Fix the NLS website of course! The most common cause of this trouble is the SSL certificate expiring on the NLS web server. NLS must be an HTTPS site, and therefore must have a valid SSL certificate. Often this cert is issued from an internal CA server to save costs, and so you don't normally get a notification when it is about to expire if you haven't put it in your calendar by hand. Make sure you keep track of this date, and replace the cert before it expires. Of course any other normal IIS or web server problem that affects the NLS website can cause this same situation, and so it is recommended that the NLS be highly available when possible. To counteract this point just a little, I have had a couple of occasions now where there was trouble validating NLS when the site was sitting behind an internal load balancer. The website seemed to be functioning properly, it would pull right up in a browser, even from the DirectAccess client computers, but the underlying **Network Connectivity Status Indicator** (**NCSI**) detection mechanism (the NLS's under-the-hood parts) wasn't validating it, and so the clients continued to think of themselves as outside the network. In both cases, as soon as we pulled the NLS website out from behind the load balancer, everything started working normally. One other cause to this behavior that I have experienced is when using the default IIS splash screen as the web page for NLS. It is common when turning on an NLS server to simply setup a new website and use the default settings, including allowing the website to use the default splash screen as its default page. For unknown reasons, in some environments this works fine, and in others NLS detection fails completely. Simply swapping out `default.htm` with a brand-new `default.htm` that contains just some simple text will resolve the issue and get NLS up and running.

I enabled NLB and DA broke!

Say you have a DirectAccess server up and running. Everything is going great, you have all kinds of users connected, and you are just loving it! So now that the technology has proven itself, it's time to turn up another DA server and create a cluster so you can make this thing redundant. So you follow all of the guides, bring the new server up to best practice standards, add the roles and walk through the wizards on the primary server to create the **Load Balanced Cluster**. The wizard is going to ask you to specify some new IP addresses, so that it can commit your existing IP addresses to be the new virtual IPs. That way, when you finish this wizard and the settings on the server side are changed, you continue to utilize the same IP addresses on the public side, so that your clients can just continue connecting like they are right now and won't have to come into the office for a Group Policy refresh before working. If the wizard utilized a new IP address for the cluster instead of your existing IP, the client-side GPOs would have to change to reflect the new IP information, and the connection would break for everyone. So this is the reason that the wizard uses the existing IPs as the VIPs, and asks you to specify new **dedicated IPs** (**DIPs**) for the server to use itself.

Back to the task at hand. You walk through the wizard, specify your new IP addresses for the server to use, click on the **Commit** button, and everything says that it completed successfully. Great! Except the phone rings. Nobody can connect from outside. You check out the server and everything is green checks, it's happy. Everything is well as far as you can tell from the server. However, if you go to an external computer and try to `telnet daserver.contoso.com 443` (which should be listening on the IP-HTTPS listener), you get a timeout. You head onto the DirectAccess server itself, and the same command results in a connection! So the server is indeed listening, but the packets are not flowing from outside to inside. But they were just a minute ago. What gives?

This issue, believe it or not, is most likely being caused by your switches. It could also be a firewall or router or even your datacenter's ISP equipment that sits between your external clients and the external NIC of the DirectAccess server, but whichever device in-line is at fault, something in that stream of devices is not allowing the packets to pass. The reason is **ARP cache**. Many network devices are "smart" enough (when you are in this situation you aren't going to think of them as very smart though) to remember the MAC address of the NIC which has been accepting these packets from the external IP address. The switch does this to speed up communications. If it can remember where packets coming in from a particular IP address have to go, it can send them straight there instead of polling the network for the right destination. When you configure NLB on a DirectAccess server, the MAC address changes. Now that there are virtual IP addresses involved which are going to be shared by two or more physical hosts, the wizards swap out the physical MAC address that has been used up until this point with a virtual MAC address. When this happens, your switch or other networking equipment may continue to send packets at the old physical MAC address, and those packets will not be received by the new configuration. So the server thinks it's listening, but the equipment in between doesn't know how to get the packets to it.

The resolution

Clear the ARP cache. You may have to dig through some documentation or reach out to your network guy to make this happen, but as soon as you flush the ARP cache on those switches or devices, they will suddenly cache the new MAC address, and packets will start flowing again. This issue is actually quite common, because most shops that are big enough to have more than one DirectAccess server are also big enough to have "smart" networking equipment. If you don't have any smart networking equipment but the symptoms line up, contact your ISP. And don't let them discount your theory! I have had to fight with ISPs more than once that this was their problem, and that simply resetting the ARP cache in their router would resolve the issue. On one case, I am certain they cleared the cache just to get rid of me because I was being so utterly persistent, and then all I could hear were crickets on their end of the line, because the traffic had magically started flowing as soon as they cleared it. It's actually quite surprising the number of networking personnel who have no idea that this cache exists.

IPv4 applications don't connect over DA

This one is near and dear to my heart, and I'll tell you the reason why in a minute. DirectAccess is based on IPv6. While you really don't have to know anything about IPv6 to run DirectAccess in your company, you do need to realize that all packets that move over the internet side of the connection, between the DA client and the DA server, are always IPv6 packets. They get wrapped up inside IPv4 so that they can make their way across the Internet, but at the core the packets themselves being generated by the laptops are IPv6 packets. Now, because the DirectAccess server contains the NAT64 and DNS64 technologies, your whole internal network can be IPv4, and that translation works all day every day, right out of the box. So none of your application servers need know anything at all about IPv6, and they are fully contactable. However, if your client-side applications, the app itself, is unable to generate IPv6 packets in the first place, that application is never going to be able to create packets that could flow over the IPv6-based IPsec tunnels, and that application will *not* connect over DirectAccess.

Sometimes, these problematic applications are simply trying to reach the wrong thing. For example, if you have an in-house, custom-coded app that is calling for an IPv4 address directly, that is not going to work over DirectAccess. I have seen many cases where an app was trying to call 192.168.1.10, for example when it tries to talk to the application server, and that is never going to work over DA. In a lot of those cases, we can simply reconfigure the client side of the application so that it calls for `Server1` instead. Once the application is configured to try and communicate with a name instead of an IP address, which name is then resolvable by the NRPT, which generates an IPv6 address from DNS64, and the application connects! If your app is hard-coded to look for an IP address and that cannot be changed, then you are out of luck. There is no existing technology that can make that app function over DirectAccess, you'll have to use an IPv4-based VPN instead for that app.

There is another situation that I run across pretty commonly, and this one is a little more complicated. Let's say you have an app that isn't working over DA, and you investigate and discover that it is indeed looking for a server name, and that name is working correctly over NRPT and the DA tunnel. In fact, you dig a little further and you can even RDP into the application server from a DA client! So you know that DirectAccess is fully able to communicate with the application server, but the application still won't open on the client computer. In this case, it is most likely that the application was coded in a way that it doesn't know how to generate IPv6 packets. Sometimes, if the software vendor still exists and is still actively coding against this application, you can go to them and ask for a new version that works with IPv6. They may have one, or you may have to wait months or years for them to create one. Sometimes, there is no hope, or you simply want it to work *now*.

App46 by IVO Networks

Up until a few months ago, I would have had to tell you the same thing as anyone else. There is no solution. But a company named **IVO Networks** has developed a utility named **App46**, and it actually resolves this issue. I had better issue a full disclaimer here, I work for IVO. This is no marketing ploy to inject it into this book; this is literally the only solution of its kind in the world. If there were others out there I would list them as well. I was part of the engineering process behind this utility, and I can vouch for its feasibility. This thing actually works. It doesn't resolve every application that can't connect over DirectAccess, but we have taken care of quite a few. As I mentioned, sometimes we find that under the hood the apps are trying to communicate with IPv4 addresses directly, and that just isn't going to happen. But as long as the app is calling for a name, we stand a good chance of intercepting this name with the App46 utility, which runs as a service on the client computers. Once intercepted, App46 takes these IPv4 packets, spins them into IPv6 packets, and sends them on their merry way across the DirectAccess tunnels.

Cannot contact some servers

Occasionally, I work with folks who have DirectAccess running, but down the road they realize that there may be a server or two that are not contactable from the DirectAccess clients for some reason. I'm not talking about the same thing as above, where the client application itself isn't working, but this would be something like a ping or RDP access to a particular server just isn't getting to its destination inside the network. There are a number of different things that could cause such behavior.

Routing

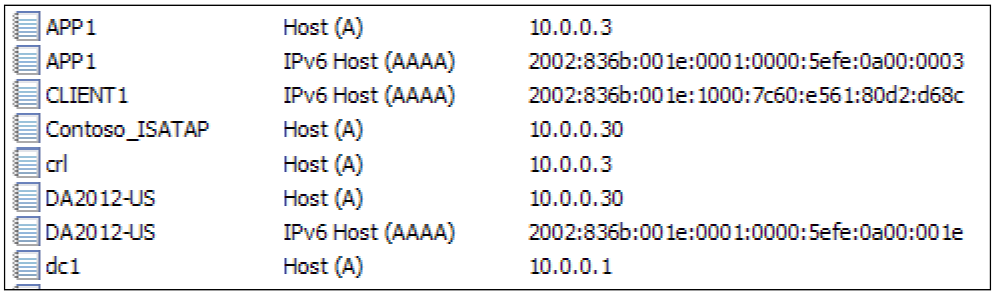
You may notice that you cannot contact a whole set of servers, and then realize that all of these servers are within the same subnet inside your network. This is the first thing I always check when working on a selective server access issue; make sure that the routes exist on the DirectAccess server. Because the internal NIC on the DA server does not have a default gateway, that means we have to build the routing table ourselves from the command prompt. If you mistyped a route or forgot to add one that will absolutely stop your DirectAccess clients from being able to get into that subnet.

Some companies use this as an access restriction method. For example, if you're a PCI compliant environment, and want to keep DirectAccess client computers out of one of your secure zones or "islands", simply keep that subnet's route out of the DA server's routing table, and they literally won't be able to get there.

To make sure the route is working, head over to the DirectAccess server and try pinging an IP address inside that subnet. If you can get through, your route is working.

Name resolution

If the route is set up properly, the next thing to check is name resolution. I am assuming here that the names of these servers you are trying to contact are in the same suffix as servers which are working over DirectAccess, in which case they should resolve to *something* when you try to ping them. If they don't resolve at all, then you might have higher level problems in your NRPT configuration. But assuming that they do resolve with ping, let's find out whether DirectAccess is trying to contact those servers via ISATAP, or with DNS64. This should be fairly easy to distinguish, because your ISATAP addresses will begin with `2002`, and the DNS64 addresses will begin with `fd` something. Let's say our server in question is named `Server1`. If you ping `Server1` from the DirectAccess client and the name resolves to an IPv6 address beginning with `2002`, then that server looks to be connected via ISATAP inside the network. If the address resolved begins with `fd`, however, then `Server1` only has IPv4 inside the network and the DNS64 mechanism on the DirectAccess server is creating that IPv6 record for you on the fly. You can also quickly check inside DNS to see what addresses are listed for `Server1`. If the `2002` address exists, then its ISATAP connected. If it only has an IPv4 address, then it's not, and DNS64 will be doing the job. Here's an example from my lab, where **dc1** is not joined to the ISATAP network, so it only has an IPv4 address in DNS, and therefore when a DA client tries to communicate with it, the DA server uses a DNS64 address. **APP1**, however, is connected to the ISATAP network, and so in DNS it has an IPv4 address and the ISATAP IPv6 address. When a DA client tries to hit **APP1**, the DA server recognizes the existence of the AAAA record, and tells the DA client to use that ISATAP address for communication.



APP1	Host (A)	10.0.0.3
APP1	IPv6 Host (AAAA)	2002:836b:001e:0001:0000:5efe:0a00:0003
CLIENT1	IPv6 Host (AAAA)	2002:836b:001e:1000:7c60:e561:80d2:d68c
Contoso_ISATAP	Host (A)	10.0.0.30
crl	Host (A)	10.0.0.3
DA2012-US	Host (A)	10.0.0.30
DA2012-US	IPv6 Host (AAAA)	2002:836b:001e:0001:0000:5efe:0a00:001e
dc1	Host (A)	10.0.0.1

There aren't typically problems with DNS64 in this area, but if the server seems to be ISATAP connected and you can't get to it, you may have to troubleshoot the actual ISATAP connection between `Server1` and the DA server. Things to look for here are to double-check all of the items we put into place in *Chapter 3*, *Configuring Manage Out to DirectAccess Clients*, make sure the custom ISATAP DNS record is resolving, that the GPO is set up correctly, that `Server1` is part of the group we created. You can head over to `Server1` and perform `ipconfig /all`, and make sure it has an ISATAP IPv6 address. If it does not, but the `2002` address exists in DNS, then the DirectAccess server is trying to send clients to the `2002` address (because DNS is telling it to go there), but the address isn't actually on `Server1`, and therefore the packets never go anywhere. Also, make sure you don't have firewalls inside your network or on `Server1` directly (think antivirus that includes a firewall) which might be blocking Protocol 41. Just like 6to4, ISATAP utilizes Protocol 41 for its transmission of packets, and if a firewall is stopping Protocol 41 between the ISATAP host and the DirectAccess server, it can cause behavior where a DirectAccess client cannot contact that ISATAP host.

Checking DNS for strange AAAA records

A few times at customer sites I have noticed unexpected AAAA (IPv6) records listed in DNS for internal servers. In some cases, a network admin setup a testing environment for IPv6 and some of the internal servers were actually configured for it. DirectAccess was trying to flow packets to those IPv6 addresses even though that IPv6 network wasn't being used, because the records existed in DNS. The DA server wasn't tapped into that IPv6 network, so the packets were going nowhere. If you have IPv6 in the network, configure the DA server to use it. If you are running IPv4-only, make sure your DNS records reflect that appropriately.

The other peculiar instance I have seen more than once are 6to4 addresses registering themselves into DNS for internal servers. In the cases where I have seen this, the internal servers actually have public IPv4 addresses assigned on one or more of their NICs. This is more common in government buildings than anywhere else that I have seen. In some cases, when an internal server has a public IPv4 address assigned, that adapter will reach out to the publically available relay for 6to4 that Microsoft has on the Internet. Therefore, the 6to4 adapter on that internal server will enable itself and establish a 6to4 IPv6 address. You can see this 6to4 address, if you log into the server (again, let's say `Server1`) and perform `ipconfig /all`. You can also see this IPv6 address listed in DNS, because typically the 6to4 adapter will choose to register itself in there.

To better understand why this screws things up, the following is the process that happens when a DirectAccess client machine tries to contact Server1:

1. The user opens an application that calls for `Server1`.
2. The DirectAccess client asks the DirectAccess server, "How do I get to Server1?"
3. DirectAccess server asks internal DNS, "How do I get to Server1?"
4. DNS responds, and if an AAAA record exists, it will choose to respond with that. In our situation here, it will respond with that 6to4 address.
5. DA server sends the 6to4 address back to the DA client.
6. DA client tries to communicate with the 6to4 address, which is not at all routable over the DirectAccess tunnels.

How to resolve this behavior? Get rid of the 6to4 addresses. If there is a firewall between `Server1` and the internet, you could block Protocol 41, and it should stop `Server1` from obtaining a 6to4 address. Otherwise, it may be easier to simply disable 6to4 on `Server1` and any other servers for which this is happening. You can do that with the same `netsh` command we talked about, or by using the same Group Policy setting that we also covered, in *Chapter 2*, *DirectAccess Environmental Best Practices*.

Does it work over IP-HTTPS and not Teredo?

Perhaps one of the most confusing ones you could encounter is a situation where your DirectAccess client computers who are connected using IP-HTTPS can reach a particular resource, but your clients that are connecting with Teredo cannot. First of all, this is a difficult conclusion to reach in the first place; it may take a considerable amount of time information gathering before you make this correlation between the transition technologies and whether or not it's working. But once you do figure that out, what on earth would make this happen? Well, Teredo likes to use ICMP. It uses "pings" in a few different ways, and it likes to have ICMP access to the resources that it is trying to contact. So if your internal network is blocking ICMP traffic, or if the local firewall on the endpoint application server is blocking ICMP, then in this case Teredo-connected clients would not be able to talk to that server, but IP-HTTPS-connected clients would be able to just fine. This happened to me in my own environment just a couple of days ago. I turned on a new server for some purpose or another, and when I tried to RDP into it from home (over DirectAccess), it kept failing. I knew that a colleague had been RDP'd into it earlier in the day, so I was sure that routing and name resolution were all working properly. Then the light bulb turned on.

I was connected using Teredo, and I checked the local Windows Firewall rules on this new server, and surely ICMP was blocked. I created a simple WFAS rule on that server to allow ICMP, and I was suddenly able to RDP into it from my Teredo-connected laptop. Because of this behavior, I recommend using GPOs to globally enable ICMPv4 and ICMPv6 on all of your internal servers in an environment where DirectAccess clients are trying to connect.

Summary

Wow, this writing experience has gone much faster than I anticipated. I must have much more hot air than was previously known. Thanks for sticking with me for the duration! I hope that this chapter was a fun ride through some personal experiences that I have encountered as it relates to fairly unique troubleshooting situations in the DirectAccess world. We have come to the conclusion of the topics that are going to be covered in this book. I sincerely hope that you enjoy DirectAccess as much as I do, and if you ever have questions or encounter interesting scenarios, I would love to hear about them! Always feel free to reach out to me directly.

Index