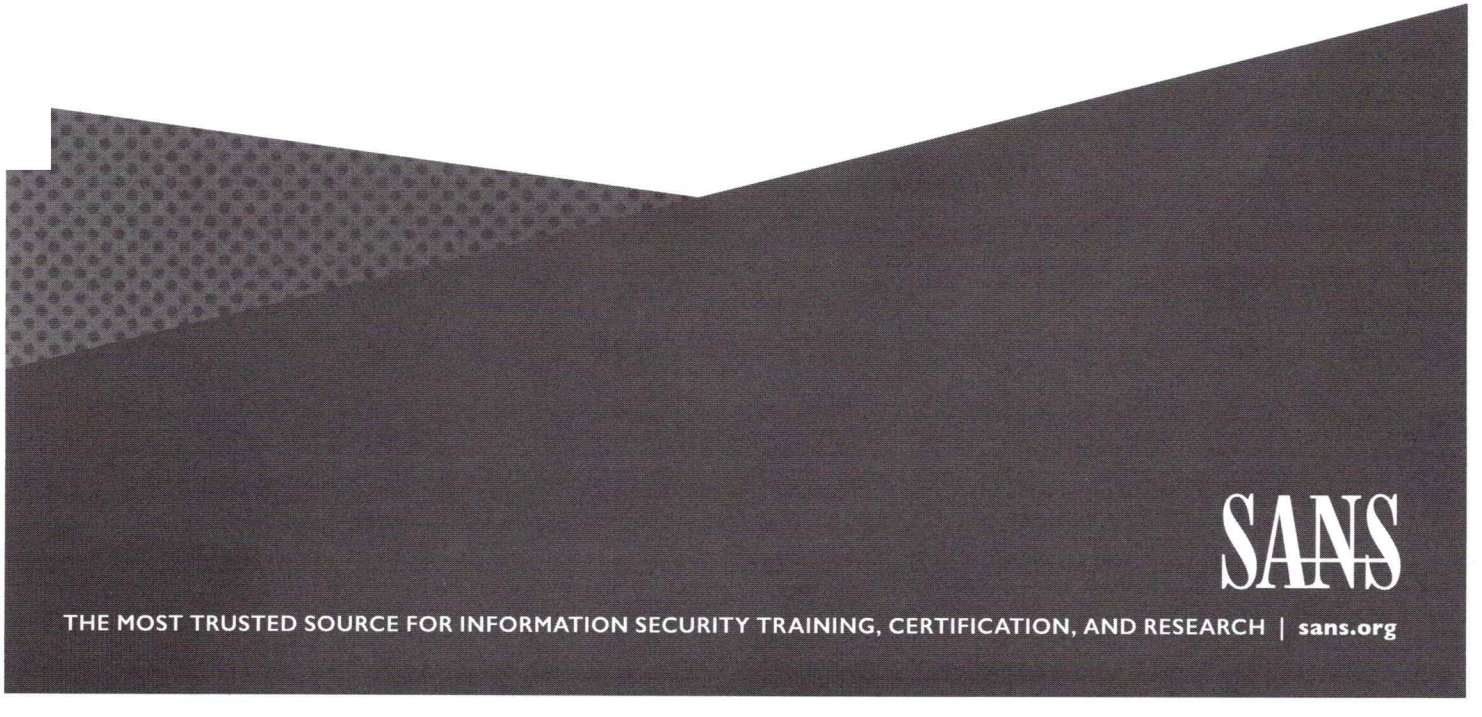
505.6

Defensible Networking and Blue Team WMI



SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

SEC505.6 | Securing Windows and PowerShell Automation

SANS

Defensible Networking and Blue Team WMI

Document Legalities

All reasonable and good faith efforts have been exerted to verify that the information in this document is accurate and up-to-date. However, new software releases, new developments, new discoveries of security holes, new publications from Microsoft or others, etc. can obviate at any time the accuracy of the information presented herein.

Neither the SANS Institute, GIAC, nor the author(s) of this document provide any warranty or guarantee of the accuracy or usefulness for any purpose of the information in this document or associated files, tools or scripts. Neither the SANS Institute, GIAC, nor the author(s) of this document can be held liable for any damages, direct or indirect, financial or otherwise, under any theory of liability, resulting from the use of or reliance upon the information presented in this document at any time.

Microsoft, MS-DOS, MS, Windows, Windows NT, Windows 2000, Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008, Windows Server 2012, Windows Server 2016, Active Directory, Internet Information Server, IIS, and .NET are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. The vomeronasal organ is an area of the human nasal cavity dedicated to detecting pheromones. Apache is a product and trademark of the Apache Software Foundation.

Other product and company names mentioned herein may be the trademarks of their owners.

The legal consequences of any actions discussed in this document are unknown. No lawyers or legal experts participated in the writing of any part of this document. Readers are advised to consult with their attorney before implementing any of the suggestions herein.

Community Document Credits

Network security is something produced by a community. Because technologies change so rapidly, the important assets are not the particular software or hardware solutions deployed today, but the skills of security administrators and their participation in the IT community. It is part of the mission of the SANS Institute to facilitate just this. This manual is a community document in that it was written with reliance upon the prior work of others and is updated regularly with the input of the security community members who use it. That means you.

If you find a significant error of fact or an important omission which would clearly add value to the document, please e-mail the contact listed below. If your suggestion is incorporated, we would be pleased to list your name as a contributor.

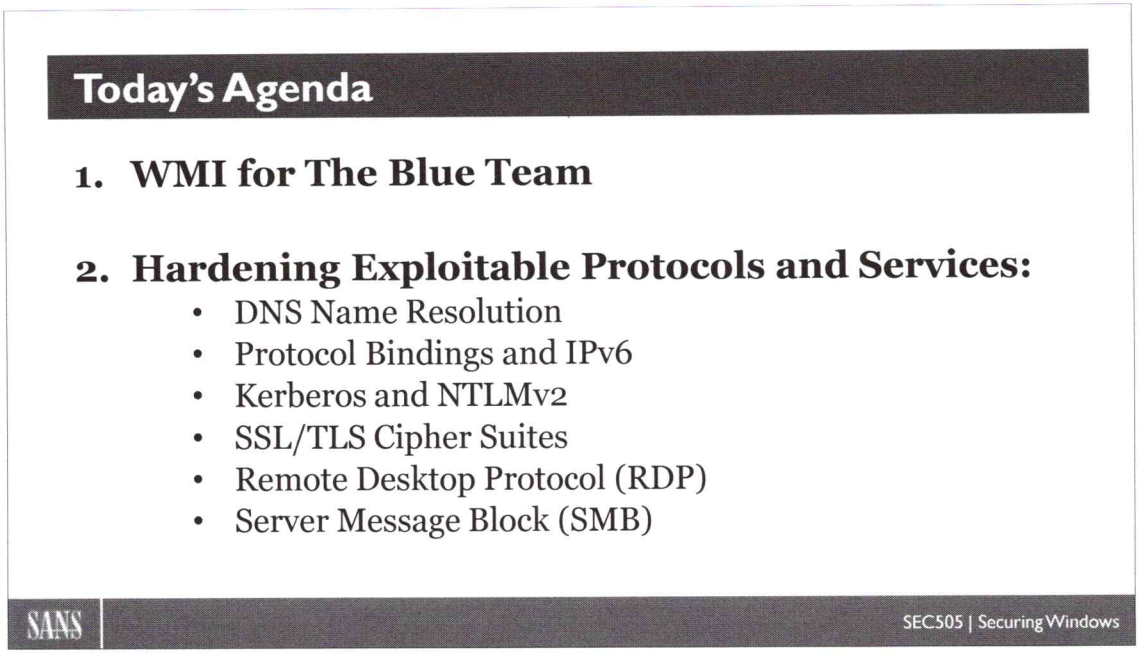
Document Author: Enclave Consulting LLC, Jason Fossen (Jason@EnclaveConsulting.com)
Document Version: 53.0 (B02_01)
Last Modified: 31.Oct.2016

Contributors:

Enclave Consulting LLC, Jason Fossen: author.
Michael Howard (www.microsoft.com): generous time spent answering IIS internals questions.
Robert Millott (ACS Defense): Brutus password-cracking utility.
Eric Neustadter (www.microsoft.com): http://corporate.windowsupdate.microsoft.com
Kathy DiGiovanni (ALLTEL Information Systems): srvinfo.exe
James Brooks (CDC): www.stbernard.com
Scott Mulcahy (Anheuser-Busch): error corrections, hardening additions.
Thierry Agassis (Unicible): IP forwarding info.
David Rhoades (www.mavensecurity.com): excellent auditing ideas and URL references.
Rick Smith (human): RDS tips and upgrade info.
Crystal Paluzzi (human): Component Services > View > Status View > see PID.
Roger Grimes (GK/PHR Holding Co.): Windows File Protection renamed file trick.
David Stevens (Carnegie Mellon): IIS Lockdown Tool info.
Gord Taylor (Royal Bank): Registry path corrections and UDL file info.
Michael Katz (Procint Security): Registry path correction.
Charles Raiford (Anheuser-Busch): SSI permissions issues.
James Damm (SAIC): ISAPI recommendations, development server tips.
Brendan Molloy (Greenwich Capital): IIS Admin service required for HTTPD.
Kurt Hinson (Tucson Electric): www.nstalker.com banner changers.
Chris Nebergall (Sandia): Mozilla authentication protocols.
Doug Brown (Sandia): NTLM/Kerberos in-channel details.
Rune Lee (Intria): Terminal Services RPC port info and other great tips.
George Mather (StVincent.org): correction to registry value path.
Frank Olmstead (US Army): burning web sites to CD-ROMs.
Rod Johnston (Somewhere in Canada): DLL to unregister.
Brett Hill (www.iisanswers.com): nothing in particular, just for having great articles.
John Millican (New Concept Tech): nice way of stopping the nbt.sys driver.
Derek Lidbom (Human): great Firefox and Kerberos info.
Jon Sweeny (Indiana Uni): numerous typos and format corrections.
Joanne Ashland (Human): SSLDigger and SSL 2.0 removal.
David Wang (Microsoft): great answers to questions about CGI restrictions.
Bob Hayden (Xerox): numerous typo/grammo/facto corrections.
Rob Ledford (City of Liberty): drive sizing changes.
Bruce Meyer (Human): swap file and memory recommendations.
Christian Prickaerts (Human): pre-forensics suggestions.
Zoher Anis (Human): pre-forensics suggestions.

Rob Lee (SANS): just for being Rob...oh, and nice suggestions too!
Alissa Torres (SANS): pre-forensics suggestions, and for not being Rob.
Philip Hagen (Human): pre-forensics suggestions.
Mike Pilkington (Human): pre-forensics suggestions.
Greg Hall (Human): fixes in this other manuals.
Rick Moffatt (Human): auditpol.exe and GPO fixes.
Tomislav Herceg (Human): mstsc /restrictedadmin for Win7
Ginny Munroe (DeadlineDriven.com): oodles of mis-scribbles -- like this line!
Rob Dunn (Human): problems with Win32_Product.

Table of Contents



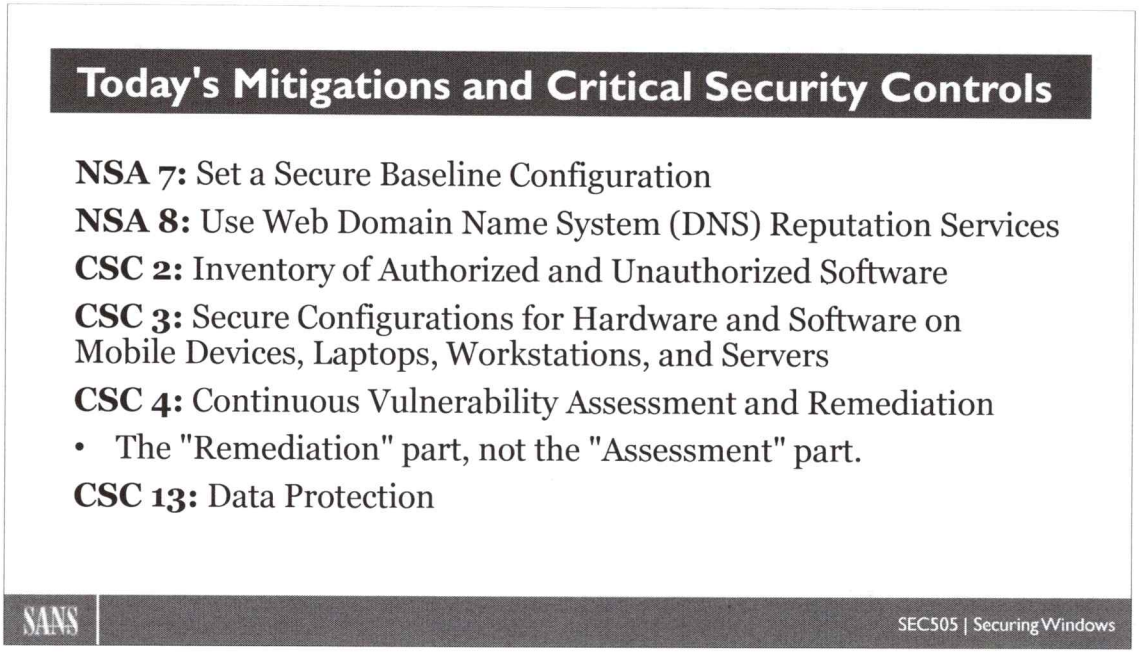
Today's Agenda

Windows Management Instrumentation (WMI) is a built-in service on Windows. WMI is Microsoft's implementation of the Distributed Management Task Force (DMTF) Web-Based Enterprise Management (WBEM). What does that mean to us? It's very powerful. Desired State Configuration (DSC) is built on top of WMI, for example. PowerShell can access WMI on local and remote machines, but the protocols used are not perfect.

There are other exploitable protocols we cannot live without either. In today's course, we will also talk about security for DNS, SSL/TLS, Kerberos, NTLMv2, Remote Desktop Protocol (RDP), and Server Message Block (SMB) protocol. The goal is to secure the ports and packets of these protocols for defensible networking, since a Windows environment more-or-less cannot function without them.

By the end of this course, you will be able to:

- Use PowerShell to access WMI on remote machines.
- Secure DNS with DNSSEC, Kerberos and sinkholes.
- Disable IPv6 tunneling features.
- Disable SSL and optimize TLS cipher suites.
- Disable LM and allow only NTLMv2 or Kerberos.
- Harden RDP against man-in-the-middle attacks.
- Encrypt and sign SMB traffic.



Today's Mitigations and Critical Security Controls

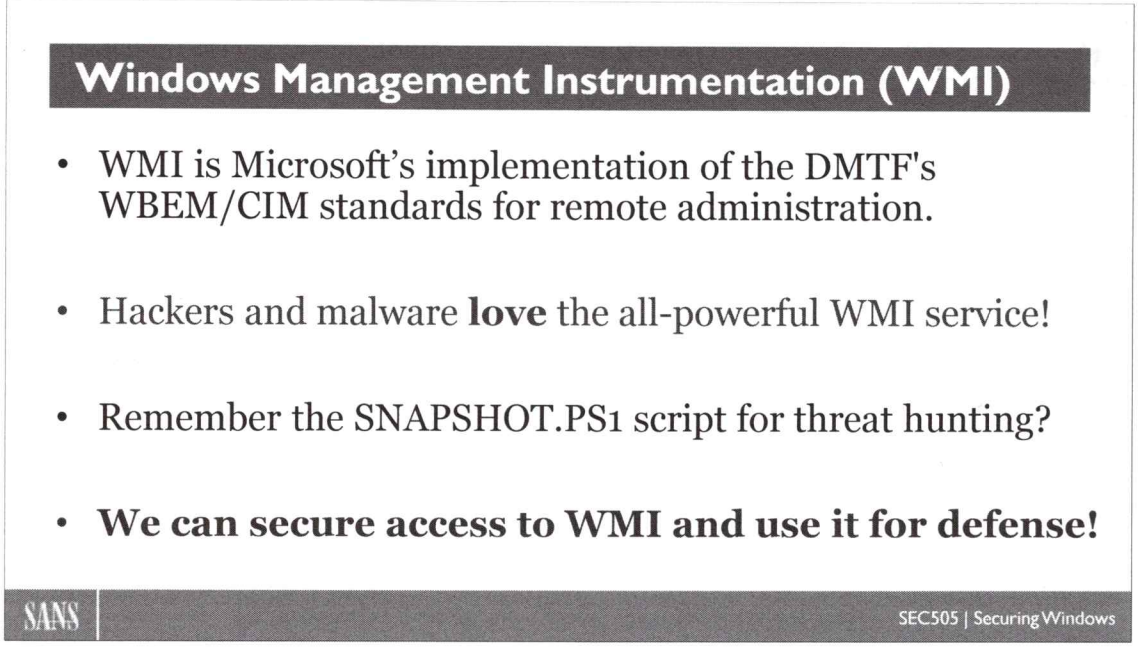
One mission of the National Security Agency (NSA) is to offer network security guidance through its Information Assurance Directorate (IAD). The NSA/IAD list of Top 10 Information Assurance Mitigation Strategies can be downloaded from the IAD web site (www.iad.gov).

The Critical Security Controls (CSC) project aims to describe the 20 most important tasks and activities for network security. You can download the latest version of the CSC from the web site of the Center for Internet Security (www.cisecurity.org).

Today's material is especially relevant for implementing the following NSA Top 10 Mitigations and CIS Critical Security Controls:

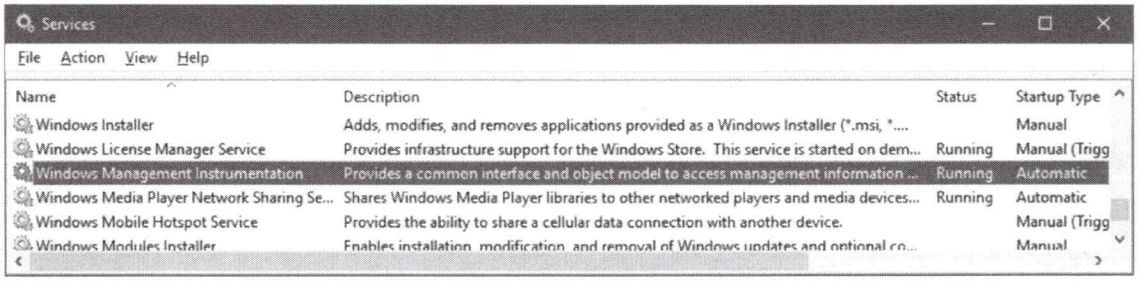
- NSA 7: Set a Secure Baseline Configuration
- NSA 8: Use Web Domain Name System (DNS) Reputation Services
- CSC 2: Inventory of Authorized and Unauthorized Software
- CSC 3: Secure Configurations for Hardware and Software on Mobile Devices, Laptops, Workstations, and Servers
- CSC 4: Continuous Vulnerability Assessment and Remediation
- CSC 13: Data Protection

But because WMI is so useful, and because we are securing absolutely essential protocols/services in Windows environments, today's material indirectly supports more NSA Mitigations and CIS Critical Security Controls than just those listed above. And for CSC 4, the emphasis here is not on the vulnerability scanning, but on the remediation.



Windows Management Instrumentation (WMI)

Windows Management Instrumentation (WMI) is the name of service found running by default on every Windows computer. The WMI service can be remotely accessed to query and change many things. Hackers and malware *love* the WMI service and have been abusing it for years!



But we cannot disable the WMI service, too much depends on it. We can, however, restrict access to the WMI service, log its abuse, and use it for good instead of evil. Remember the SNAPSHOT.PS1 script we use for pre-forensics and threat hunting? Many of the commands in that script use the WMI service, but that's just the tip of the iceberg of what's possible. Learning how to use WMI is an essential PowerShell skill.

What Is WMI?

Windows Management Instrumentation (WMI) is Microsoft's implementation of the Distributed Management Tasks Force's (DMTF) Common Information Model (CIM) for Web-Based Enterprise Management (WBEM) and remote administration.

OK, so what does *that* mean? Let's unpack these terms.

What Is WBEM?

Remember SNMP? The goal of SNMP was to provide a vendor-neutral protocol for getting and setting configuration information on any networking device. Each SNMP-capable device would have one or more standardized miniature databases, called Management Information Bases (MIBs), which SNMP could query and write. Well, WBEM is similar to the next generation of SNMP, but with a *much* more ambitious goal.

The goal of WBEM is to be able to monitor and manage hardware devices, operating systems, applications, services, directories, users, or any other network-attached entity, no matter what variety of vendors have supplied these entities to an organization, and to do all this with standardized protocols that even make the monitoring/management systems independent of vendor and location.

What Is The DMTF?

The Distributed Management Task Force (DMTF) is the group designing the standards and protocols to make WBEM a reality. The DMTF wants a WBEM world. The DMTF is composed of frustrated industry leaders (IBM, Cisco, Novell, Microsoft and 180 other organizations) who all fought for years to have their own proprietary remote-management standards dominate the marketplace, but now they've seen that the enormous complexity of a WBEM world will require cooperation if anything is to be achieved at all. In a sense, they've learned from TCP/IP-- it's better if we all just get along.

What Is The CIM?

WBEM defines goals, not implementation details. The Common Information Model (CIM) is the core set of standards for implementing WBEM. Roughly speaking, CIM is to WBEM like RFCs are to TCP/IP.

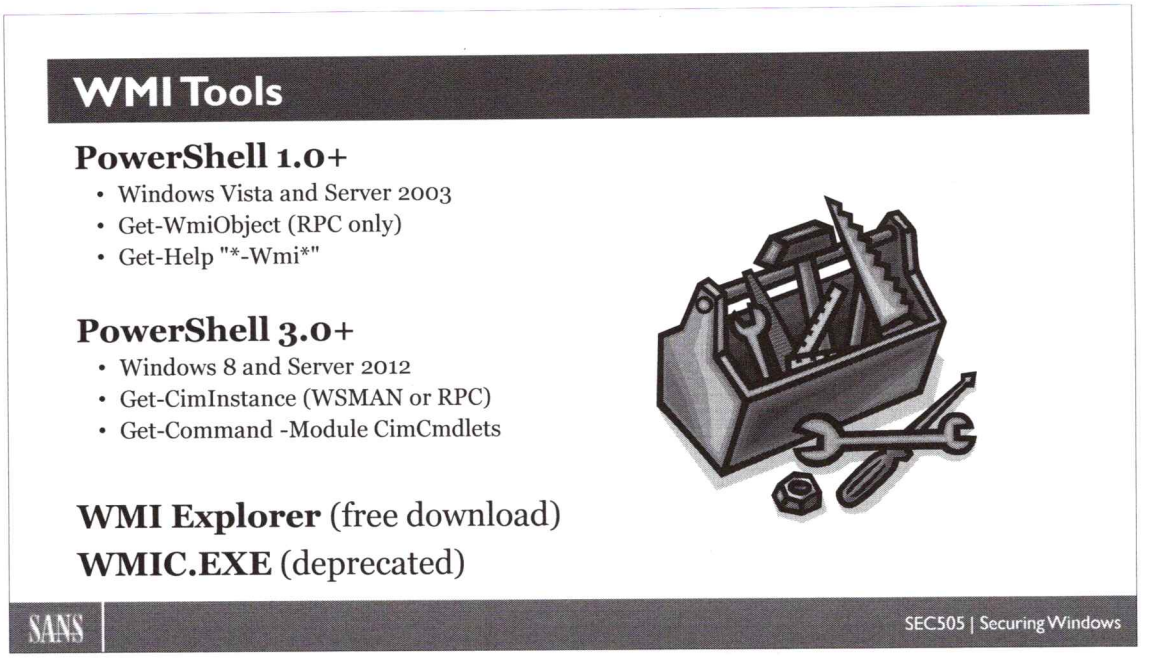
Most importantly, CIM defines how data should be formatted and named. Each CIM-compliant platform will have a service which can receive and understand remote CIM-formatted requests and commands, and reply with CIM-formatted data and messages. For example, CIM organizes data into a hierarchical namespace, similar to the way Active Directory organizes user/computer information into a hierarchical namespace. Different types of hardware and software will always be found in certain well-known locations in the hierarchy. And you will be able to search and enumerate the CIM namespace on a remote box when you don't know what to ask for.

Windows Management Instrumentation (WMI) is Microsoft's implementation of CIM for Microsoft's operating systems, services and applications. WMI is "MS-CIM", but not *embraced and expanded CIM* in proprietary ways, it's just CIM on Microsoft boxes.

What Can I Do With WMI?

WMI can be used to manage event logs, shutdown/reboot machines, execute commands on remote systems, query NTFS permissions on files, enumerate services for auditing, manage shared folders, list processes and their paths, enumerate network connections, add DNS records, and *many* other tasks. PowerShell itself is just one part of what Microsoft calls the "Windows Management Framework (WMF)", and WMI is another

major part of the WMF. PowerShell and WMI are normally upgraded together when installing a new version of the WMF.



WMI Tools

There are many tools related to Windows Management Instrumentation (WMI).

PowerShell 1.0 WMI Cmdlets

PowerShell 1.0 was first installed by default on Windows Vista and Server 2003. It included several WMI-related cmdlets, the most important of which is Get-WmiObject:

- Get-WmiObject
- Remove-WmiObject
- Invoke-WmiMethod
- Set-WmiInstance
- Register-WmiEvent

To list your WMI-related cmdlet names:

```
Get-Help "*-wmi*"
```

When these original WMI cmdlets connect to a remote computer, they use DCOM RPC.

PowerShell 3.0 CIM Cmdlets

PowerShell 3.0 was first installed by default on Windows 8 and Server 2012. PowerShell 3.0 and later include newer cmdlets named after the Common Information Model (CIM), the most important of which is Get-CimInstance:

- Get-CimInstance

- Get-CimAssociatedInstance
- Get-CimClass
- Get-CimSession
- Invoke-CimMethod
- Set-CimInstance
- New-CimInstance
- New-CimSession
- New-CimSessionOption
- Register-CimIndicationEvent
- Remove-CimInstance
- Remove-CimSession
- Import-BinaryMiLog
- Export-BinaryMiLog

To list your WMI-related CIM cmdlets:

```
Get-Command -Module CimCmdlets
```

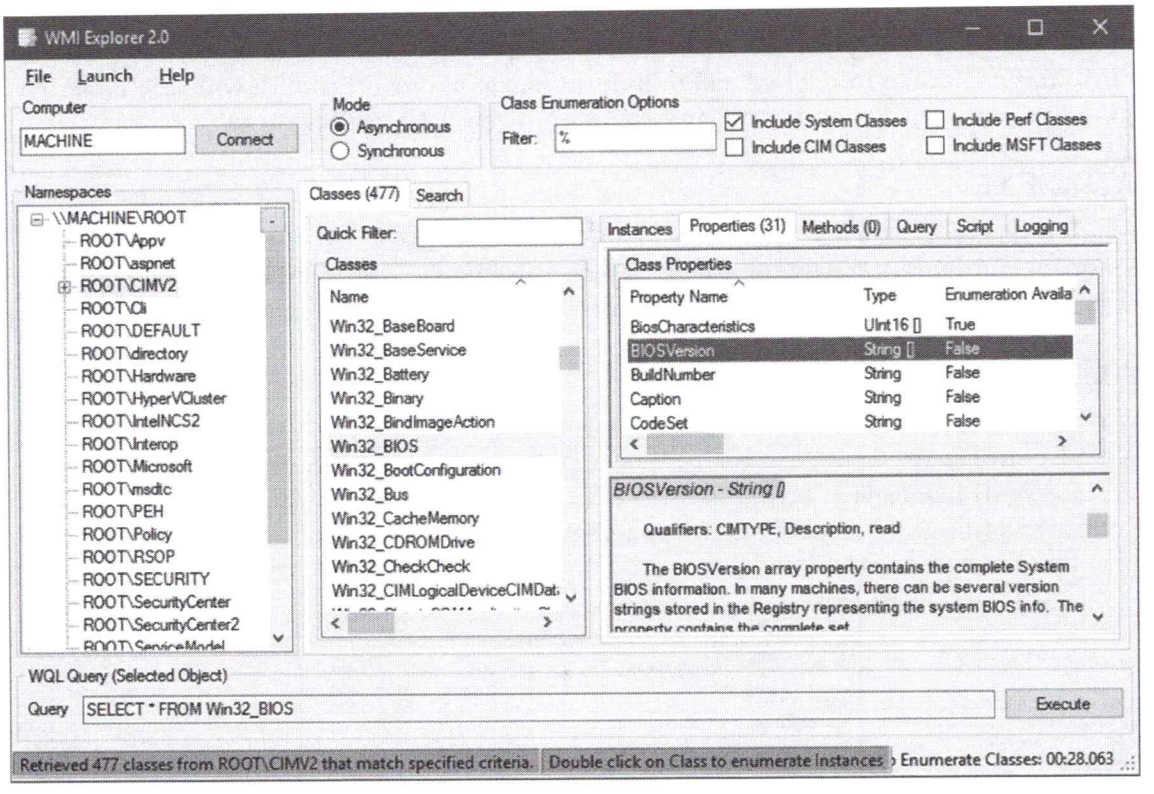
These newer CIM cmdlets are easier to use and faster than the original WMI cmdlets (like Get-WmiObject). The newer CIM cmdlets also comply better with the CIM and Web Services Management (WSMAN) standards, which offers better cross-platform compatibility. The newer CIM cmdlets can also use the WSMAN protocol, not just DCOM RPC.

Fortunately, the syntax and use of the CIM cmdlets is often very similar to the original WMI cmdlets, especially when using WQL queries. Here is a table for reference which maps the original WMI cmdlets to the newer CIM cmdlets.

Original WMI Cmdlet	Newer CIM Cmdlet
Get-WmiObject	Get-CimInstance
Get-WmiObject -List	Get-CimClass
Remove-WmiObject	Remove-CimInstance
Invoke-WmiMethod	Invoke-CimMethod
Set-WmiInstance	Set-CimInstance
Set-WmiInstance –PutType CreateOnly	New-CimInstance
n/a	Register-CimIndicationEvent
n/a	Get-CimAssociatedInstance

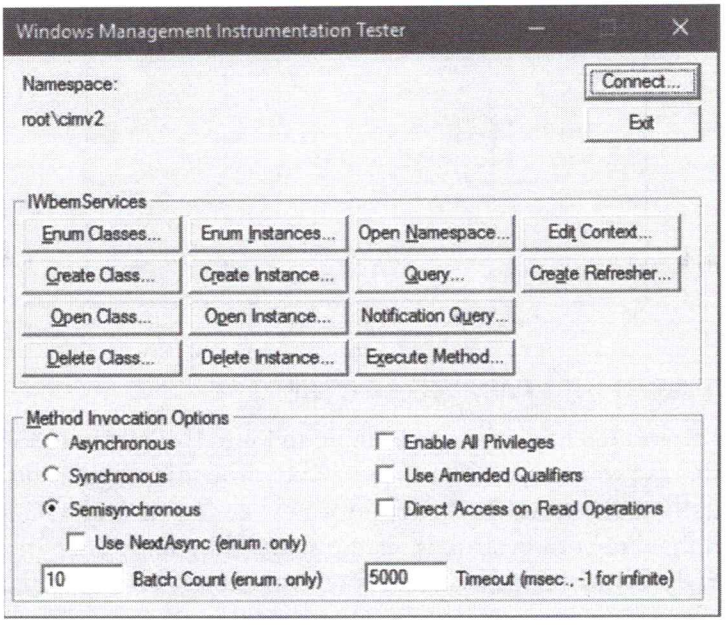
WMI Explorer (Free Download)

WMI Explorer is not a built-in tool, but it is a free download (https://wmie.codeplex.com). WMI Explorer is a very nice, easy-to-use, graphical tool for browsing WMI namespaces, classes, properties and methods. It can also connect to remote computers and generate sample PowerShell code for interacting with WMI.



WBEMTEST.EXE

WBEMTEST.EXE is a built-in graphical tool to interact with the WMI service on local or remote machines. It's only advantage over WMI Explorer is that it is already built in (C:\Windows\System32\wbem\wbemtest.exe).



CIM Studio (Free Download)

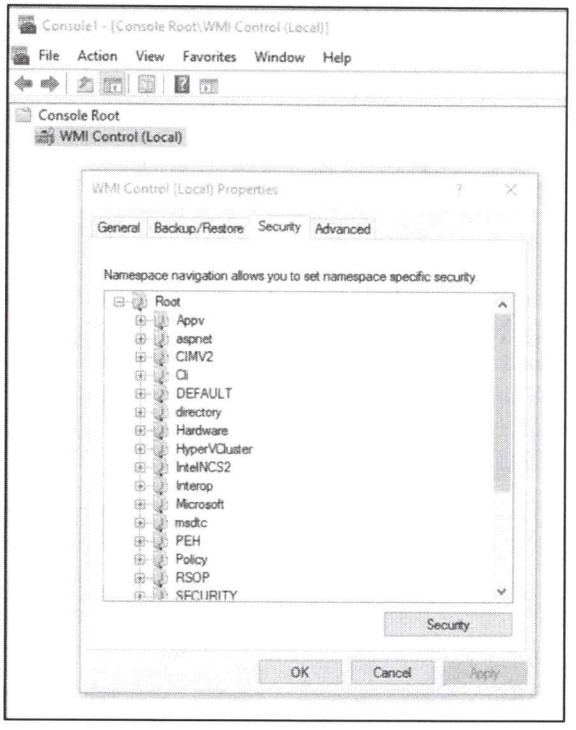
CIM Studio is also a free, graphical tool (http://msdn.microsoft.com/downloads/), but it's not quite as polished or fast as WMI Explorer or even WBEMTEST.EXE.

WMIC.EXE

Windows includes a WMI command-line tool named "WMIC.EXE" that can be used to get or set configuration data for a wide variety of settings. There are many WMIC.EXE examples on the Internet. However, this tool has been deprecated in favor of PowerShell.

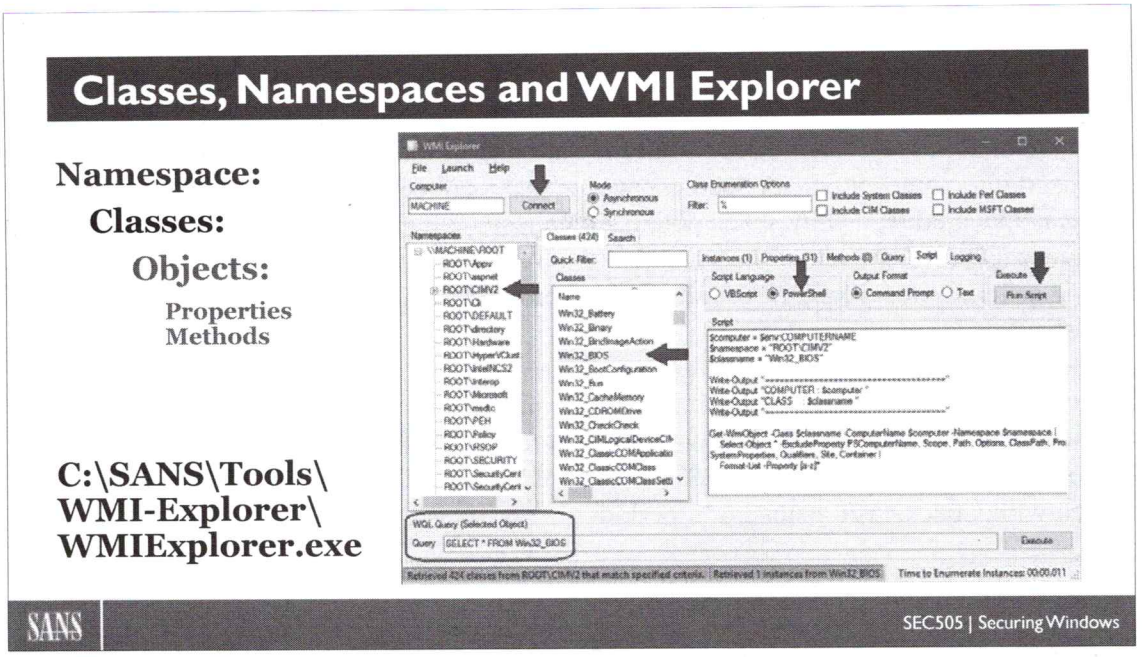
WMI Control (MMC.EXE snap-in)

WMI Control is the name of a snap-in that can be added to an MMC.EXE console. The WMI Control snap-in can target a local or remote computer, and is mainly used to manage WMI namespace permissions. To see these permissions, right-click on the WMI Control snap-in itself, select Properties, and go to the Security tab.



[WMI], [WMICLASS] and [WMISEARCHER]

WMI objects can be created by casting a variable to [WMI], [WMICLASS] or [WMISEARCHER] as "solution accelerators." But while this might require fewer keystrokes, it actually makes it more difficult for new coders to understand how to use WMI. The notion that "fewer keystrokes" is the same thing as "easier" is incorrect. This manual will generally avoid using these and focus on using the -Query parameter instead.



Classes, Namespaces and WMI Explorer

WMI information is structured around objects, objects are collected into classes, and classes are organized under namespaces.

WMI Classes

A "class" in WMI/CIM is a set of one or more objects which represent other things in Windows; for example, there is a class named "Win32_Process" which contains objects representing processes, and a "Win32_BIOS" class with just one instance object whose properties contain information about the firmware on a computer.

By analogy, in economics we might talk about "the middle class" as a set of people within a certain income range. In general, a class is a set of things, and we can talk about each thing individually (like "Justin McCarthy is a member of the middle class") or we can talk about the class as a whole (like "the income of the middle class increased by two percent last year").

Think of a class as a table in a database, where each object in the class is like a record in that table, and the properties of those objects correspond to the fields (or column headers) of those records. Hence, a class is like a database table, and an object in a class is like a record in a row of that table.

To list the classes available in a particular namespace, such as "root\CIMv2":

```
get-wmiobject -query "select * from meta_class"
  -namespace "root\cimv2"
```

The get-wmiobject cmdlet also has a "-list" switch to view classes.

WMI Namespaces

There are hundreds of WMI classes, but they are not all lumped together. Classes are organized into namespaces. A "namespace" is just a collection of classes. By analogy, if a class is like a table, a namespace is like a database with many tables/classes inside of it. A namespace does not actually correspond to a real folder or database, it's just an abstract naming scheme. Just like DNS domain names do not physically exist, neither do WMI namespaces.

The topmost WMI namespace is named "root", just like the topmost DNS domain, and the most important namespace under root is named "CIMV2", which is short for CIM version 2. Unlike DNS, though, the path to CIMV2 would be written as "Root\CIMV2", written with a backslash instead of a period.

To see a list of all the WMI namespaces:

```
get-wmiobject -query "select * from __namespace"
  -namespace "root" | format-table Name
```

But what do these classes contain? How can we dump their contents?

WMI Explorer

To help visualize and browse your namespaces and classes, you can download a free, graphical, easy-to-use tool named "WMI Explorer" from https://wmie.codeplex.com.

In your training VM, you can also find the WMI Explorer tool here:

```
C:\SANS\Tools\WMI-Explorer\WMIExplorer.exe
```

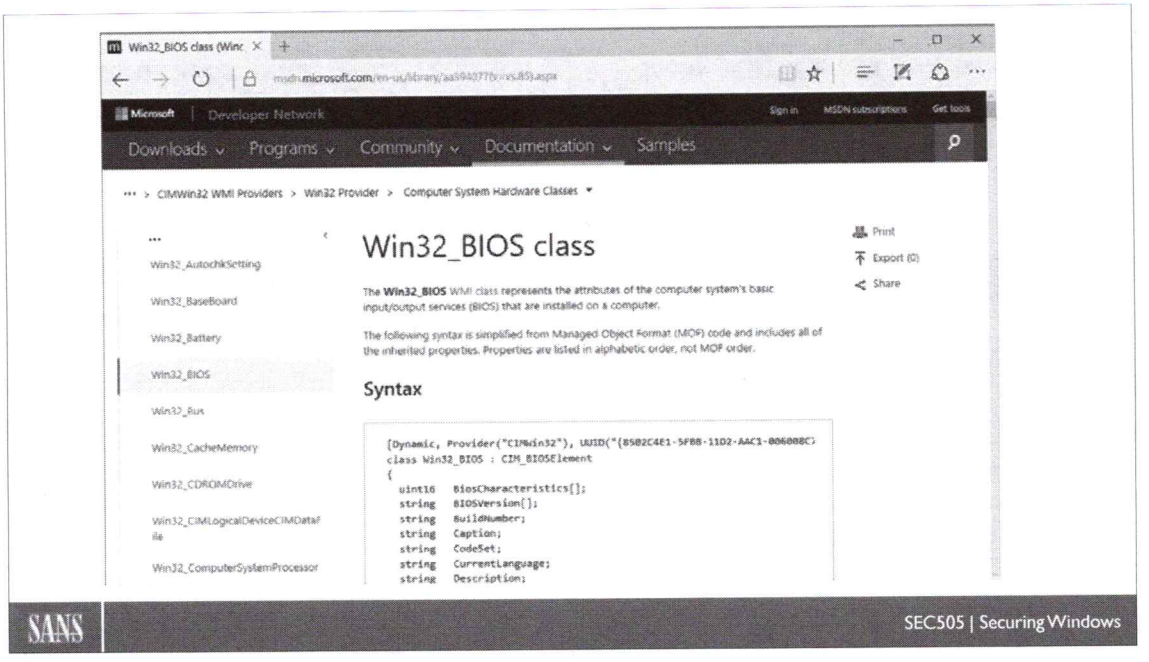
As shown in the slide, in the WMI Explorer graphical interface, click the Connect button to list the namespaces on a local or remote computer (defaults to your own computer).

In the list of namespaces on the left, double-click the ROOT\CIMV2 namespace, wait a few seconds, then notice in the middle pane the list of classes from that namespace.

In the list of classes in the middle, scroll down to the Win32_BIOS class and double-click it. On the right-hand side you can now browse the tabs to see the instances, properties and methods of the objects from the selected class.

On the Script tab, choose PowerShell, then click the Run Script button. A new PowerShell window appears and executes the script code shown. Very handy!

Finally, in the bottom left-hand corner, WMI Explorer shows the WQL query executed to gather the data displayed in the tool. This is similar to an SQL query of a table in a database, except that here the table is a WMI class.



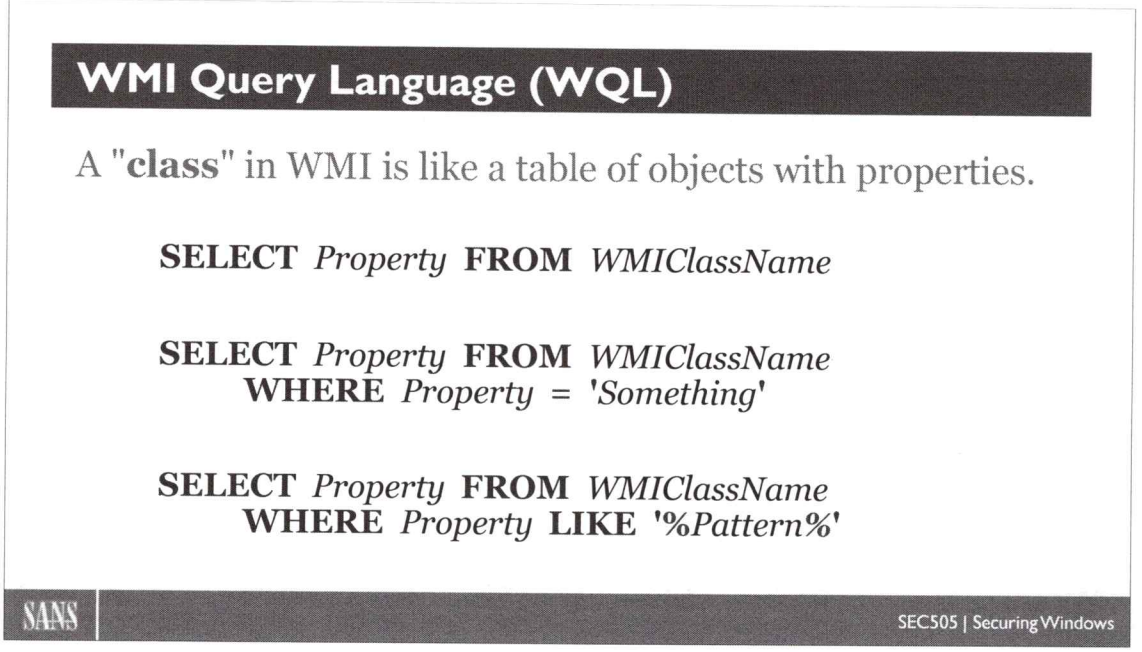
Search On WMI Class Name

There are too many WMI classes, properties and methods to memorize. Very often, a property has only a numeric code as a value, but what do these numbers mean?

Use any Internet search engine and do a search on a particular WMI class name, such as "Win32_BIOS", and often the very first link returned will take you to Microsoft's web site with the documentation on that class. The on-line documentation includes a description of the class, the mapping of property code numbers to their meanings, the effects of calling various methods, links to related classes, miscellaneous remarks, OS requirements, and sometimes code samples written in C# or PowerShell too.

So, when running commands like the following, please don't feel lost and overwhelmed by the strange-looking output: there is lots of on-line documentation. Just search on the WMI class name and property names you're interested in!

```
$bios = Get-WmiObject -Query "SELECT * From Win32_BIOS"

$bios | Format-List *
```



WMI Query Language (WQL)

WMI Query Language (WQL) is a simplified subset of ANSI standard Structured Query Language (SQL) for databases. With WQL you can query the classes in the CIM in a consistent, understandable, explorable way. Learning basic WQL is also a good investment in one's IT career because of its carryover to SQL with database applications and servers.

The format of a typical WQL query looks like this:

```
SELECT <property> FROM <class> WHERE <property> = 'foo'
```

Or, when using the "LIKE" keyword to do pattern matching:

```
SELECT <property> FROM <class> WHERE <property> LIKE '%foo%'
```

The keywords "SELECT", "FROM", "LIKE" and "WHERE" are very similar to SQL. Typically, some or all properties are selected from a class in the CIM as though that class were a table in a database. A property can be visualized as the column header names in a table, with each row or record in that table being a separate object. To select all properties/columns, select the asterisk ("*") instead of specific property names.

The "WHERE" clause in a WQL query is optional, just as using the where-object cmdlet is optional in a pipeline; both have the same purpose, namely, to limit which properties or objects are returned by the query or pipeline.

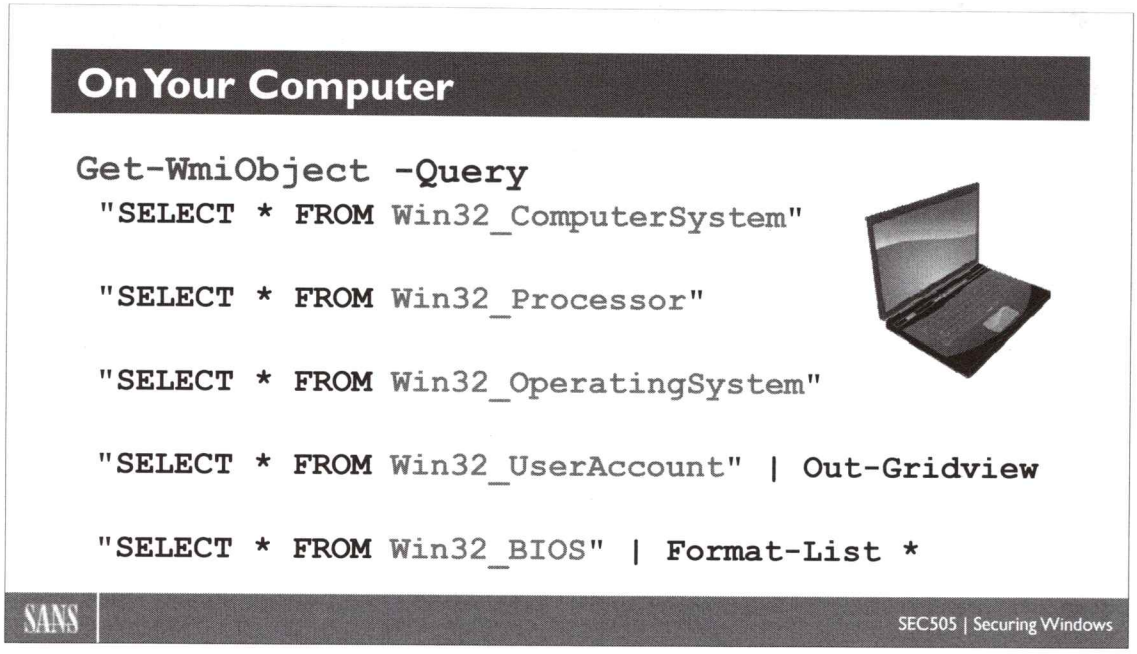
When using the "LIKE" keyword to do pattern matching, here are the valid wildcards:

Wildcard	Matching Data
%	Similar to "*", matches any string of zero or more characters
_	Similar to "?", any single character (use square brackets for literal)
[...]	Replace "..." with a set or range of characters, like [a-m] or [ione]
[^...]	Replace "..." with a set or range of characters to NOT match
[a=e]	The "=" indicates a range, so [a=e] is the same as [abcde]

There are other special operators and keywords in WQL too (similar to PowerShell):

Operator or Keyword	Meaning
=	Equal (-eq)
!=	Not Equal (-ne)
<>	Not Equal (-ne)
<	Less Than (-lt)
>	Greater Than (-gt)
<=	Less Than or Equal (-le)
>=	Greater Than or Equal (-ge)
IS NULL	Equal To Nothing/Blank/Empty (-eq $null)
IS NOT NULL	Not Equal to Nothing/Blank/Empty (-ne $null)
(...)	Logical Grouping
AND	Boolean And (-and)
OR	Boolean Or (-or, which is not XOR)
TRUE	Boolean True ($true)
FALSE	Boolean False ($false)

The names of the classes in WMI are rather strange looking, but don't let that throw you off, they are similar in concept to tables in a database. We will see examples in the pages which follow.



On Your Computer

Please run the commands as shown in the slide above. Notice that every command is the same except for the last word, hence, you can use the up-arrow button on your keyboard to edit your last command.

To see more information, run the commands in the slide above and pipe the output through "Format-List *" such as:

```
Get-WmiObject -Query "SELECT * FROM Win32_BIOS" | Format-List *
```

If many objects are returned from a query, pipe into Out-Gridview for easy searching:

```
Get-WmiObject -Query "SELECT * FROM Win32_Process" | Out-Gridview
```

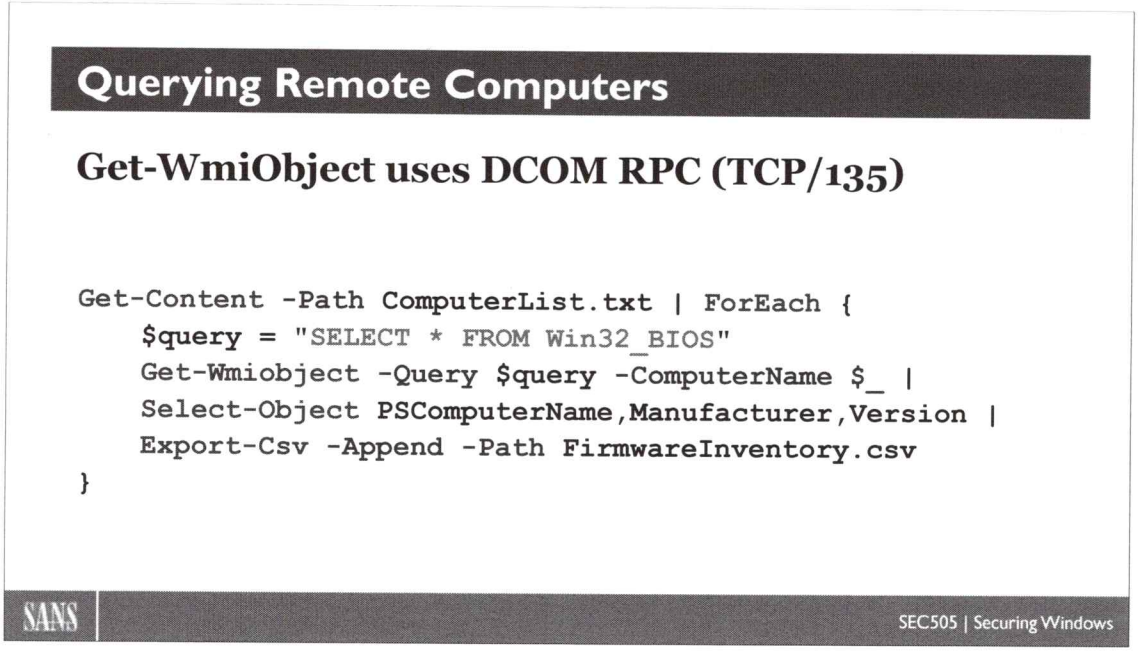
You can execute a command and capture its output to a variable, then extract each desired property by name from the variable; for example:

```
$bios = Get-WmiObject -Query "SELECT * From Win32_BIOS"

$bios.Manufacturer

$bios.Version
```

Notice that the commands in the slide do not use the -Namespace parameter. The CIMv2 namespace is the default namespace PowerShell uses if you do not specify one.

Finally, it is possible to pipe output through Select-Object to strip away all WMI object properties except those you wish to see:

```
get-wmiobject -query "SELECT * FROM Win32_QuickFixEngineering" |
  select-object PSComputerName,InstalledOn,HotFixID
```



Querying Remote Computers

WMI is designed to work locally or remotely. Every one of the examples above can be used to query a remote computer as long as you are a member of the local Administrators group on that computer. WMI is a DCOM RPC-based protocol, so the firewall on the remote system must also allow inbound access to TCP/135 and whatever upper-level ephemeral TCP port number is also negotiated by the RPC endpoint mapper (TCP 49152-65535).

To connect to a remote computer and perform a WMI query:

```
$box = "server3.sans.org"

get-wmiobject -query "SELECT * FROM Win32_BIOS" -computer $box
```

WMI will use integrated Windows authentication by default, meaning that it will automatically try to use Kerberos first, then NTLM, to authenticate to the remote system with the single sign-on credentials of the person running the script.

PowerShell 3.0 CIM Cmdlets And WSMAN Over HTTPS

PowerShell 3.0 and later includes a newer set of cmdlets which are preferred:

```
get-module -name cimcmdlets | select -expand exportedcommands
```

These newer CIM cmdlets are easier to use and faster than the original WMI cmdlets.

The newer CIM cmdlets comply better with the DMTF's CIM and Web Services Management (WSMAN) standards, which offers better cross-platform compatibility.

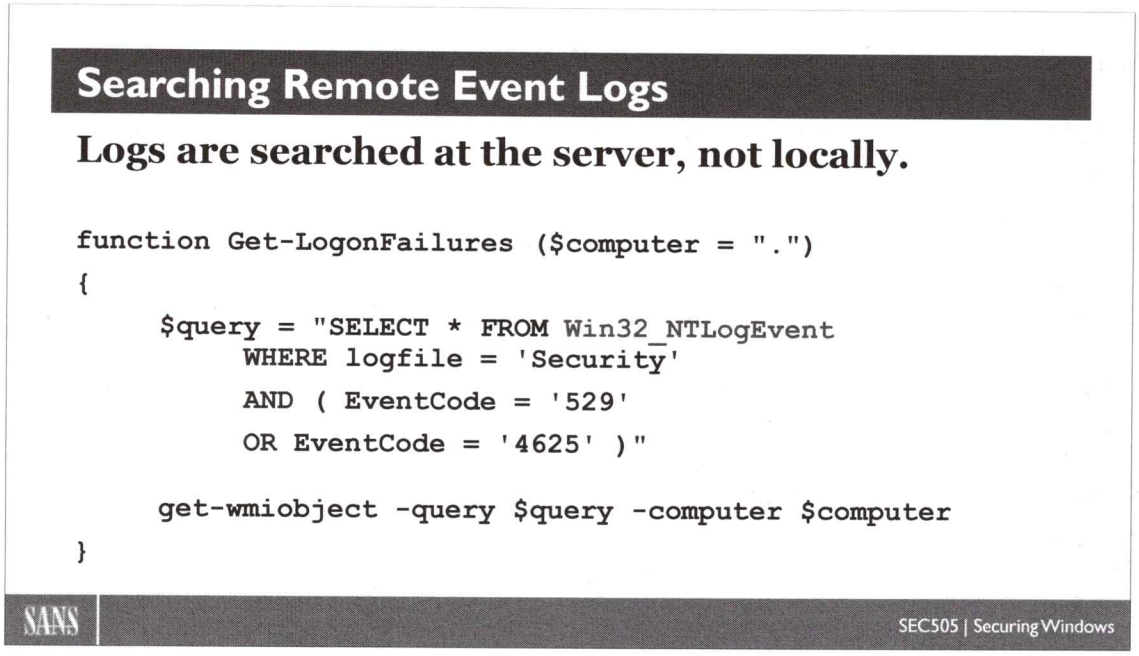
The older WMI cmdlets can only use RPC DCOM to connect to remote systems, while the newer CIM cmdlets can also use the WSMAN protocol, just like PowerShell remoting; in fact, with the newer CIM cmdlets, WSMAN is the default, DCOM must be explicitly chosen. And, just like with PowerShell remoting, if an SSL/TLS certificate is installed and configured at the target, the WSMAN connection for WMI can be run through HTTPS too. This makes the CIM cmdlets available for use over the Internet.

However, the CIM cmdlets require PowerShell 3.0. Only Server 2012, Windows 8 and later come with PowerShell 3.0 or later by default, not Windows 7 or Server 2008 R2. To get PowerShell 3.0 or later on Server 2008 R2 and Windows 7 requires installing Service Pack 1 or later, installing .NET Framework 4.0 or later, and installing Windows Management Framework (WMF) 3.0 or later. If you're going to upgrade PowerShell, then might as well install the latest version of WMF.

> **Note:** Until Windows 10 or later replaces Windows 7 on the majority of workstations managed by the attendees of this course, this manual will continue to show examples using the original WMI cmdlets.

Fortunately, the syntax and use of the CIM cmdlets is very similar to the original WMI cmdlets, especially when using WQL queries. Here is a table for reference which maps the original WMI cmdlets to the newer CIM cmdlets.

Original WMI Cmdlet	Newer CIM Cmdlet
Get-WmiObject	Get-CimInstance
Get-WmiObject -List	Get-CimClass
Remove-WmiObject	Remove-CimInstance
Invoke-WmiMethod	Invoke-CimMethod
Set-WmiInstance	Set-CimInstance
Set-WmiInstance –PutType CreateOnly	New-CimInstance
Register-WmiEvent	Register-CimIndicationEvent
Get-WmiObject -Query <associates>	Get-CimAssociatedInstance



Searching Remote Event Logs

WMI can also be used to query local or remote event logs. Importantly, the query is given to the remote computer, the WMI service at the remote computer extracts the data requested, and sends only that data across the network. This is much faster than downloading the entire log and performing the search locally. An event log on a domain controller might be 20GB in size, but we might need only the failed logon events, which might only be 10MB of data.

When searching an event log, we can use the WHERE keyword in an WQL query with multiple selection criteria to precisely define the data we want. This reduces the amount of data which must be returned over the network.

To query a local or remote event log for bad username/password logon events:

```
# C:\SANS\Day6-Servers\WMI\WMI_EventCode.ps1

function Get-LogonFailures ($computer = ".")
{
   $query = "SELECT * FROM Win32_NTLogEvent WHERE logfile =
      'Security' AND (EventCode = '529' OR EventCode = '4625')"

   get-wmiobject -query $query -computername $computer
}
```

The output of the above function will be an array of objects representing event log messages. These objects have several properties (such as User, TimeGenerated, EventCode) that correspond to the same information seen in the graphical Event Viewer

snap-in. We can perform further filtering using these properties and extract just the property data that we need.

```
$events = Get-LogonFailures

$events.Count     #Total number of events returned.

$events | Select TimeGenerated,CategoryString,Type,EventCode
```

InsertionStrings and Message Properties

Some of the data you'll want is found in the InsertionStrings and Message properties of the event log objects. The InsertionStrings property is an array of strings, hence, we can access each string individually by its index number; for example:

```
$example = $events[0]

$example.InsertionStrings

$first  = $example.InsertionStrings[0]

$second = $example.InsertionStrings[1]
```

The Message property is the "payload" of the event log entry. It is one long string, not an array, hence, we'll need to carve out the substrings we want using regular expressions or other string-cutting tricks. The Select-String cmdlet can use regular expression patterns to carve out substrings (see C:\SANS\Day1-PowerShell\Extract-Text.ps1 and .\Regular-Expressions.ps1). There are several free tutorials on regular expressions on the Internet, the topic cannot be covered here.

DMTF Date and Time Format

WMI usually encodes date and time information in Distributed Management Task Force (DMTF) format. A date and time in Distributed Management Task Force (DMTF) format looks similar to "20171127131906.000000-300", which can be interpreted as follows:

Year	Month	Day	Hour	Minutes	Seconds	UTC Offset in Minutes
2017	11	27	13	19	06.000000	-300

WMI and other services often use DMTF-formatted timestamps, so it is useful to be able to convert back-and-forth between DMTF and more human-friendly formats.

Many WMI objects have script methods added by PowerShell to convert to/from these special fields from/to something more humanly readable (such as *System.DateTime* objects).

Here are a couple functions to convert DMTF times when you otherwise run across them and must convert by hand:

```
function Convert-DMTFtoDateTime ($DMTF)
{
  [Management.ManagementDateTimeConverter]::ToDateTime($dmtf)
}


function Convert-DateTimeToDMTF ($Date)
{
  [Management.ManagementDateTimeConverter]::ToDmtfDateTime($Date)
}
```

For example, if you run the following command:

```
Convert-DMTFtoDateTime -DMTF '20171008121410.180726-000'
```

The output is a System.DateTime object, the same type of object created by the Get-Date cmdlet, with lots of human-readable properties.

On Your Computer

This lab has multiple parts.

Using Embedded Help In Scripts

Switch to the C:\SANS\Day6-Servers\WMI folder:

```
cd C:\SANS\Day6-Servers\WMI
```

Review the built-in help from the comments of the Get-SuccessfulLogon.ps1 script:

```
Get-Help -Full .\Get-SuccessfulLogon.ps1
```

Open the script in ISE to see the comments at the top used for generating help text:

```
ise .\Get-SuccessfulLogon.ps1
```

There are a variety of keywords (Synopsis, Description, Parameter, etc.) which match the sections in the output of "Get-Help -Full *ScriptName.ps1*". Each keyword must be preceded by "#." and no space character before the keyword.

Close the script tab in ISE, we just wanted to see the special help comments.

Querying Event Logs With WMI

Use WMI to query all logon events in the past 48 hours from a local or remote computer:

```
.\Get-SuccessfulLogon.ps1 -WithinLastHours 48 -AllLogons
```

> **Note:** If the query takes too long or you get an error message about a quota violation, do a Ctrl-C to cancel, then try again with fewer hours (perhaps 8).

View interactive logons in a table (includes both console and remote desktop logons):

```
.\Get-SuccessfulLogon.ps1 -WithinLastHours 48 -InteractiveLogons|
 Out-GridView
```

View the last 10 network logons with source computername or IP address:

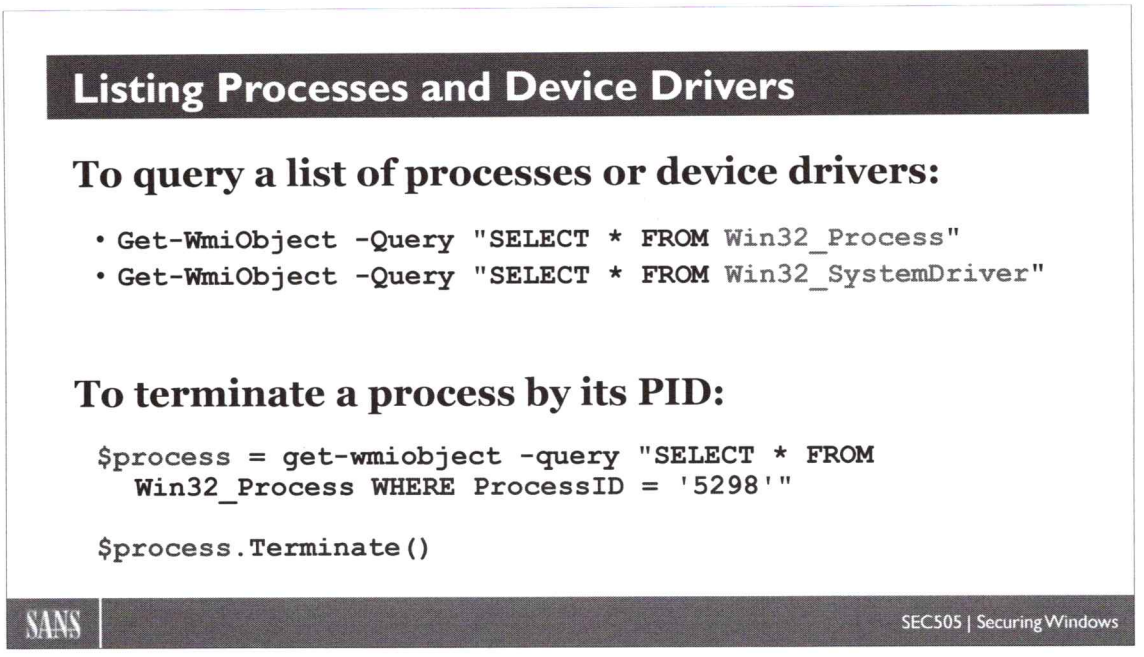
```
.\Get-SuccessfulLogon.ps1 -WithinLastHours 48 -NetworkLogons |
 Sort-Object DateTime | Select-Object -Last 10 | Out-Gridview
```

> **Note:** The above query might return nothing on your test VM. Depending on the authentication method, the source computer or IP address might not get logged.

Export all logon events to a CSV file, then view only the events for admin users:

```
.\Get-SuccessfulLogon.ps1 -WithinLastHours 48 -AllLogons |
  Export-Csv -Path Logons.csv

Import-Csv -Path Logons.csv | Where { $_.User -Like "*admin*" } |
  Out-GridView
```

> **Note:** If your username does not match "*admin*", view the CSV file and edit the search string so that it will match at least one username in the text.



Listing Processes and Device Drivers

To obtain device driver information from a local or remote computer:

```
# C:\SANS\Day6-Servers\WMI\Get-DriverWithWMI.ps1

function Get-DriverWithWMI ($computer = ".")
{
   get-wmiobject -query "SELECT * FROM Win32_SystemDriver" `
    -computername $computer |
   select-object Name,DisplayName,PathName,ServiceType,`
         State,StartMode
}
```

To obtain a list of running processes from a local or remote computer:

```
# C:\SANS\Day6-Servers\WMI\Get-ProcessWithWMI.ps1

function Get-ProcessWithWMI ($computer = ".")
{
   get-wmiobject -query "SELECT * FROM Win32_Process"
         -computername $computer |
   select-object Name,ProcessID,CommandLine
}
```

After querying the processes on a remote computer, the list of processes could be filtered by process name or some other characteristic. The matching list of processes could then all be terminated.

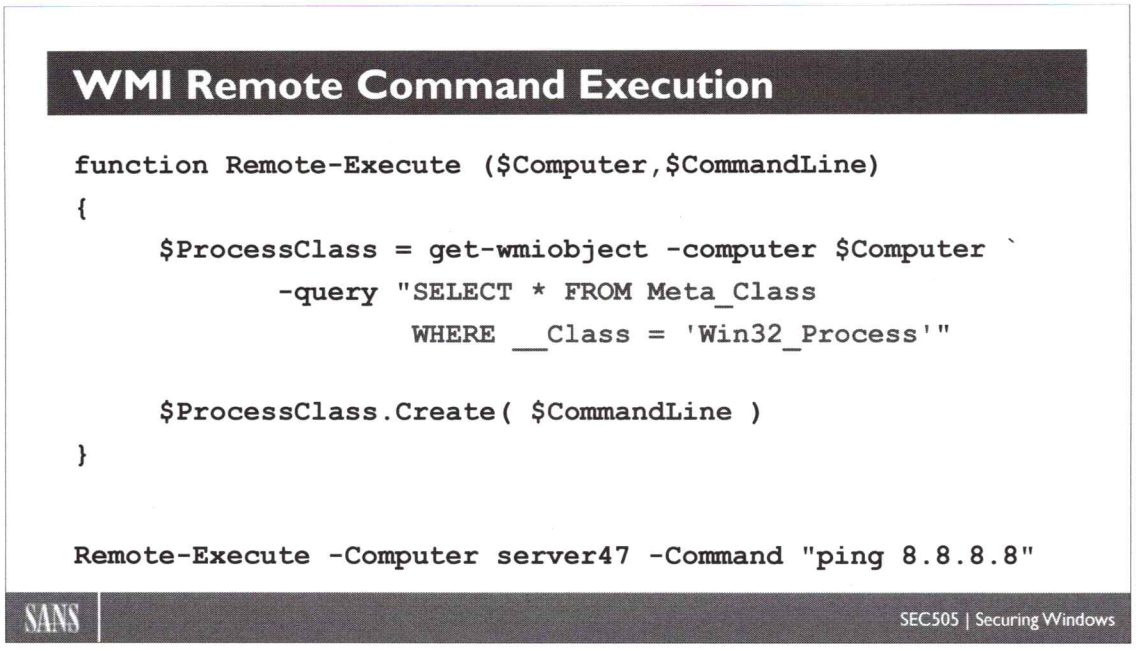
To terminate a process on a local or remote computer:

```
# C:\SANS\Day6-Servers\WMI\Terminate-ProcessWithWMI.ps1

function Terminate-ProcessWithWMI ($computer, $ProcessID)
{
  $process = get-wmiobject -query "SELECT * FROM
      Win32_Process WHERE ProcessID = '$ProcessID'" `
      -namespace "root\cimv2" -computername $computer

  $results = $process.Terminate()

  if ($results.ReturnValue -eq 0) { $true } else { $false }
}
```

When we call the Terminate() method on a remote process object, how do we know it actually worked? The output is captured to the $results variable, which has a property named ReturnValue. The ReturnValue will be the exit code number produced when the WMI service tried to terminate the process at the remote computer. If the ReturnValue is zero, then it worked, the process was terminated; if the ReturnValue is a non-zero number, then the attempt failed. The non-zero code number could be used to research on the Internet the reason for the failure (do a search on "windows system error codes"). Here are a few of the common exit code numbers:

0 : Success
2 : Access Denied
3 : Insufficient Privilege
9 : Not Found

Could the above function be enhanced to map the ReturnValue code numbers to human-readable text to aid with troubleshooting? No problemo!



WMI Remote Command Execution

A WMI class is not only a collection of instances of that class, but an object itself with its own properties and methods. The *Win32_Process* class, for example, contains objects representing running processes on the computer, but the class itself has methods to create and destroy instances contained within it, i.e., to launch and terminate processes.

The *Win32_Process* class is an instance of the *Meta_Class* class, hence, we can query *Meta_Class* to get a reference to the *Win32_Process* class itself. And once we have *Win32_Process* as an object in our script, we can use it to launch new processes. In other words, we can use it for remote command execution.

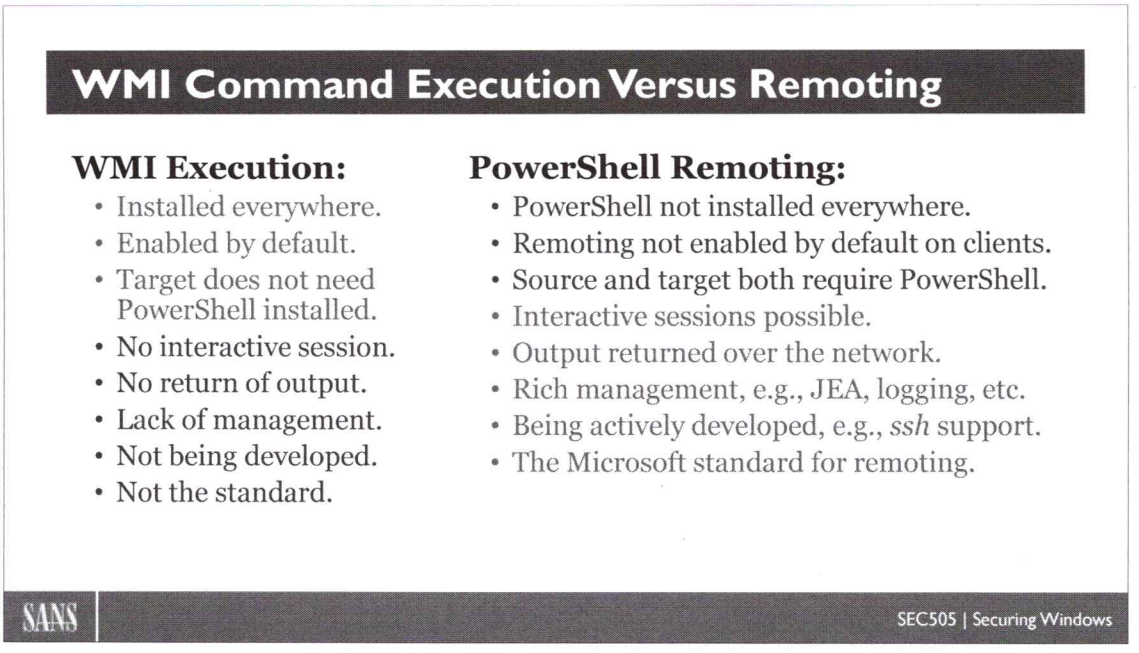
To launch a process on a local or remote computer:

```
# C:\SANS\Day6-Servers\WMI\Create-ProcessWithWMI.ps1

function Remote-Execute ($computer, $commandline)
{

 $Win32_Process_Class = get-wmiobject -query "SELECT * FROM
  Meta_Class WHERE __Class = 'Win32_Process'"
  -computername $computer

 $results = $Win32_Process_Class.Create( $commandline )

 if ($results.ReturnValue -eq 0) { $results.ProcessID }
 else { $false }
}
```

How do we know if it worked? Calling the Create method on the Win32_Process class outputs an object with a ReturnValue property, and that property has the Windows system exit code number for attempting to call the method. If the ReturnValue is zero, the method worked, the process was created; if the ReturnValue is a non-zero number, the attempt failed for some reason. The above function returns the process ID number of the remote process if it worked, or $false if it didn't work.

Hackers and malware love using the WMI service for remote command execution because the WMI service is enabled by default.

For us on the Blue Team, how does this compare with PowerShell remoting? Which should be used on a daily basis?



WMI Remote Command Execution Versus PowerShell Remoting

Executing commands on remote computers through WMI is similar to PowerShell remoting, but there are some important differences:

- The WMI service is installed and enabled on all Windows machines by default, even on stand-alones. PowerShell remoting is not enabled by default on all Windows computers (yet).

- To use WMI for remote command execution, the target computer does not need to have PowerShell installed. PowerShell remoting requires PowerShell to be installed both locally and on the target computer.

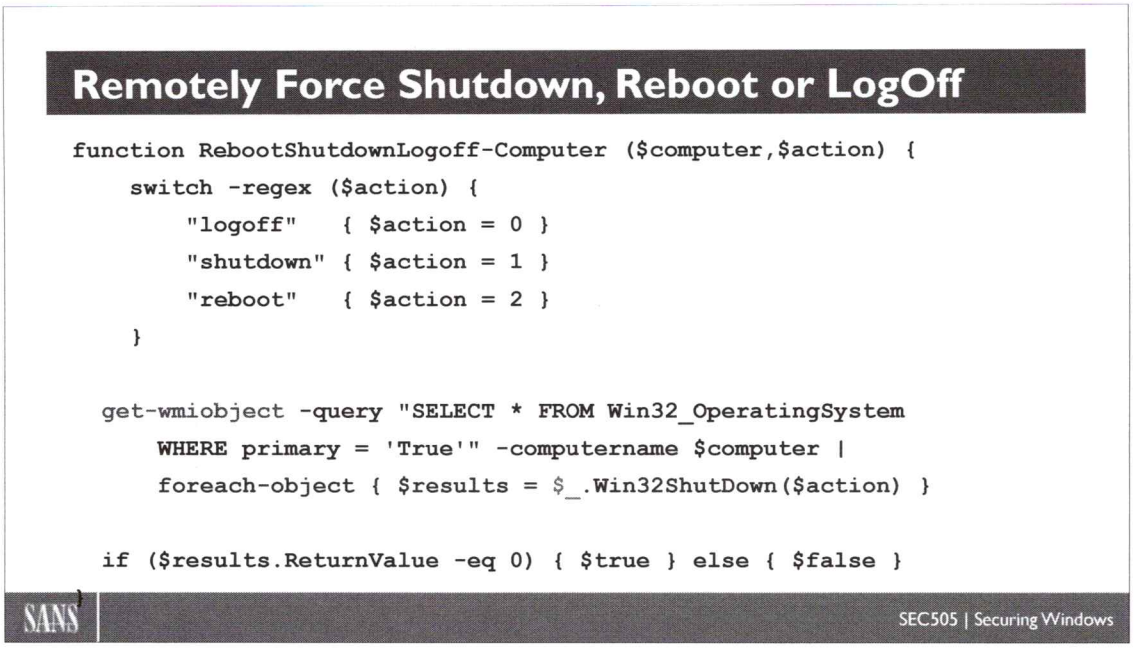
- WMI remote command execution does not provide an interactive session (it's not like telnet), it just tries to execute the command and then disconnects. With the Enter-PsSession cmdlet, on the other hand, PowerShell remoting does offer an interactive session when needed.

- WMI remote command execution launches a new process at the target computer, but the output of this process, if any, is not streamed back to the local computer. If you want to capture the output of a command run through WMI, the typical trick is to redirect the output to a temp file somewhere, get the file, parse the file locally, then delete the file. PowerShell remoting streams the output of commands back to the local computer automatically. This output can be captured to an array or piped into further commands, no temp file is necessary.

- PowerShell remoting supports other advanced features like constrained environments, fan-in remoting, fan-out remoting, parallel execution over many targets, easy backgrounding as a job with collation of returned data, etc. WMI as a whole has nice management features, but for remote command execution specifically it is relatively crude or simplistic by comparison to PowerShell remoting.

- Microsoft is committed to expanding and improving PowerShell remoting. It is the gold standard for remote command execution on Windows. WMI as a whole is being actively improved by Microsoft, but WMI remote command execution specifically is a feature that has barely changed in a decade.

When there is a choice, it is better to use PowerShell remoting.



Remotely Force Shutdown, Reboot or LogOff

To shutdown, reboot or log off the interactive user on a remote computer:

```
# ..\WMI\RebootShutdownLogoff-Computer.ps1

function RebootShutdownLogoff-Computer ($computer, $action) {
    switch -regex ($action) {
        "logoff"             { $action = 0 }
        "forced.*logoff"     { $action = 4 }
        "shutdown"           { $action = 1 }
        "forced.*shutdown"   { $action = 5 }
        "reboot"             { $action = 2 }
        "forced.*reboot"     { $action = 6 }
        "powerdown"          { $action = 8 }
        "forced.*powerdown"  { $action = 12 }
    }

 get-wmiobject -query "SELECT * FROM Win32_OperatingSystem `
     WHERE primary = 'True'" -computername $computer `
     -namespace "root\cimv2" |
    foreach-object { $results = $_.Win32ShutDown($action) }

 if ($results.ReturnValue -eq 0) { $true } else { $false }
}
```

How would we know which arguments to the -Action parameter are acceptable? Fortunately, we can view the embedded help in the comments in the script:

```
Get-Help -Full .\RebootShutdownLogoff-Computer.ps1
```

From the help, we see that only the following arguments to -Action are supported:

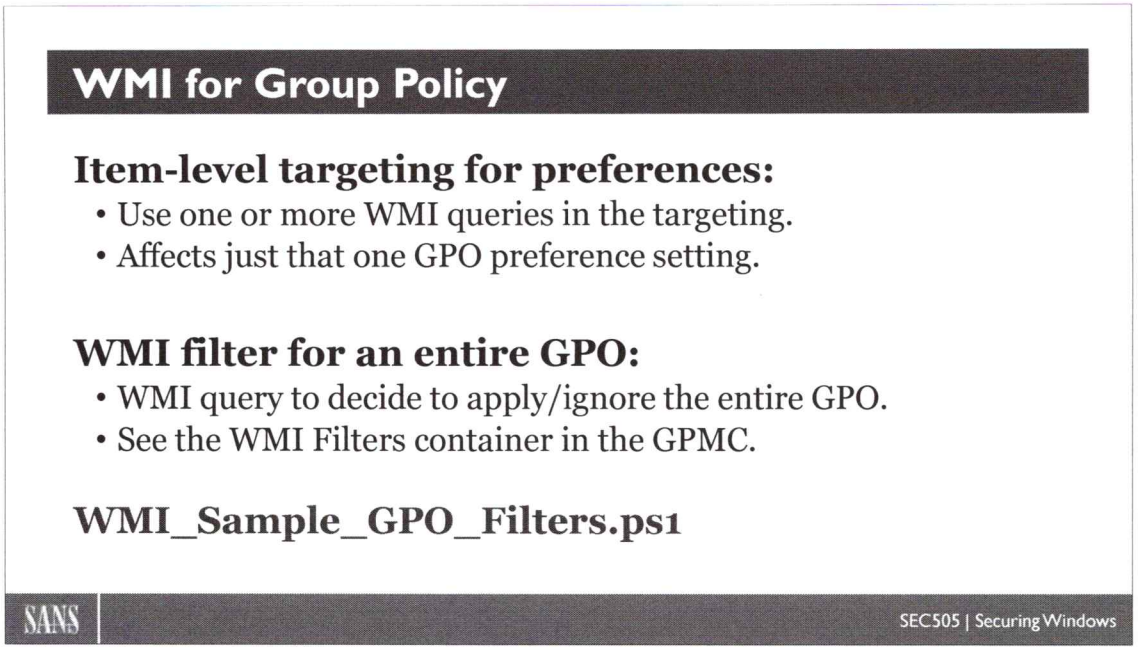
- logoff
- shutdown
- reboot
- powerdown
- forced logoff
- forced shutdown
- forced reboot
- forced powerdown

A "forced" reboot or shutdown means that any unsaved changes will be lost if a user is logged on locally and has open applications. What the user will see is that all of his or her applications simply terminate without warning, the user will be logged off, then the machine reboots or shuts down.

> **Note:** With an Advanced Function using the [CmdletBinding()] decoration, it would be possible to actually enforce the above list of permitted arguments, but there is not enough time to cover Advanced Functions in today's course.

One hard thing to understand in the function is how the output of the Get-WmiObject command is being piped into ForEach-Object. Remember, anything piped into ForEach-Object will go into the "$_" variable. In this case, what is being piped is a single object, an object representing the running operating system of the remote computer (the "primary" OS is the running OS, but a dual-boot computer could have another "non-primary" OS too).

When you press Ctrl-Alt-Del on your keyboard and select either Reboot or Shutdown, you are invoking a method in the low-level Windows API named "Win32ShutDown", and this is the same function called by our PowerShell function. This is a big deal! PowerShell can call many Windows API functions through WMI and other .NET Framework features, such as P/Invoke.



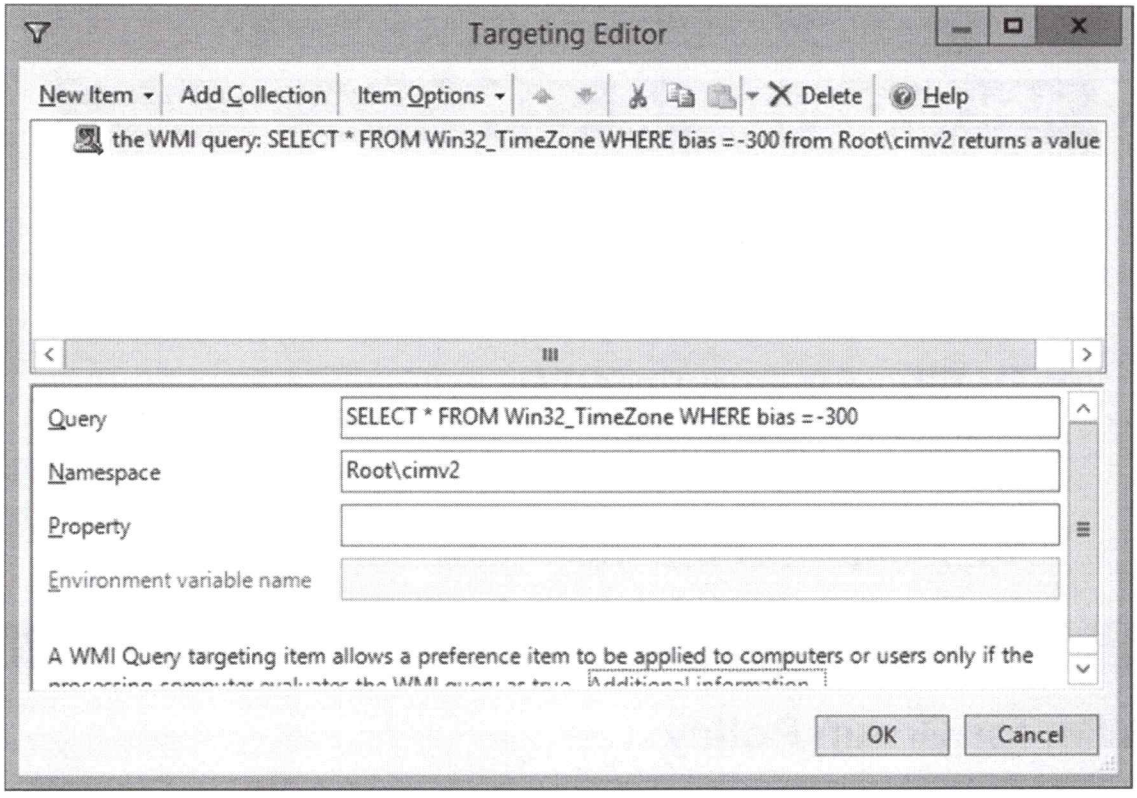
WMI for Group Policy

A Windows Management Instrumentation (WMI) query can extract information about processes, drivers, users, shares, software versions installed, patches, printers, log data, ports, network adapter cards, IP configuration, registry values, and more. A PowerShell script can talk to the WMI service to extract this data or reconfigure these settings, but WMI can also be utilized by Group Policy directly without PowerShell.

A WMI query can be included as part of a GPO to control changes made by the GPO. This can be done in two ways: 1) a WMI query can be used with item-level targeting to control the application of an individual preference setting in a GPO, or 2) a WMI query can be associated with the GPO as a whole to decide whether or not the GPO should be applied at all. The difference is between using a WMI query to filter the application of one preference in a GPO, though the rest of the GPO may still be processed, and preventing the GPO as whole from being processed.

GPO Preference Item-Level Targeting

The Preferences container in a GPO has settings that can be filtered based on a large variety of criteria. One of these criteria is named "WMI Query". This was discussed in the section on GPO Preferences.



The dialog box above can be seen by opening any GPO > editing any setting under Preferences for User or Computer Configuration > Common tab > check "Item-level targeting" > Targeting button > New Item menu > WMI Query.

Paste your WMI query into the "Query" field. If this query returns anything, it is considered a true result, and the associated preference setting is applied. If the query returns nothing, this is considered a false result, and the associated preference setting is not applied.

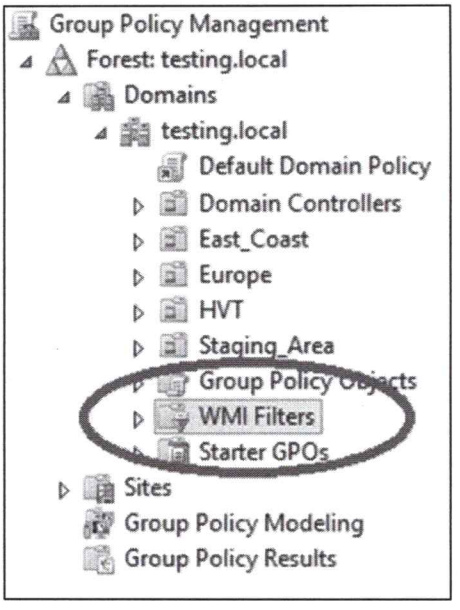
(The "Namespace" field identifies the WMI namespace which contains the WMI class you wish to query; it is almost always going to be "Root\CIMv2". The "Property" field is optional and will rarely be used; it can take the value of a WMI property and assign it to a variable.)

Whole GPO WMI Filter

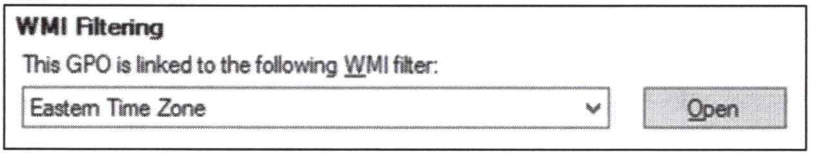
A WMI query can also be used to decide whether the GPO as a whole should be applied or ignored. This is prior to and independent of any item-level targeting for GPO Preferences. Using WMI to decide whether to apply/ignore the entire GPO is called "GPO Filtering" or "WMI Filtering", depending on who you ask. This feature requires that the computer processing the GPO (not the domain controller) be Windows XP, Server 2003, or later.

Try It Now!

- To configure a WMI filter on a GPO, write your WMI query and copy it to the clipboard. Next, open the Group Policy Management console > expand your domain > right-click the WMI Filters container > New > enter a name for the filter > Add button > paste in your query > OK > Save.



- To assign this filter to a GPO, click that GPO to select it > go to the Scope tab on the right > pull down the list of WMI Filters at the bottom > select the one you want > Yes.



In this example WMI query/filter, we could test whether the computer is in the Eastern time zone, i.e., it is 300 minutes behind Greenwich Mean Time, and, if it is, apply the GPO, or if it is not, ignore the GPO:

```
SELECT * FROM Win32_TimeZone WHERE bias =-300
```

If you think this looks like SQL, you're correct. It's actually "WMI Query Language (WQL)", but WQL was loosely modeled on SQL, so if you've got database management experience then you've got a good head start on understanding WQL.

WMI Query Language (WQL) Examples

Here are several WMI query examples (WMI_Sample_GPO_Filters.ps1) which could be used to decide whether to apply a preference setting or to apply an entire GPO.

An WMI query can use the following pattern-matching symbols with the LIKE operator:

%	Similar to "*", any string of zero or more characters
_	Similar to "?", any single character (use square brackets for literal)
[...]	Replace "..." with a set or range of characters
[^...]	Replace "..." with a set or range of characters to NOT match

```
#Applies if Server 2012 Standard Edition is the operating system:
SELECT * FROM Win32_OperatingSystem WHERE Caption = 'Microsoft
Windows Server 2012 Standard'


#Applies if the OS is for a workstation or client (not a server):
SELECT * FROM Win32_OperatingSystem WHERE ProductType = '1'


#Applies if the OS is NOT for a client OS (perhaps for a server):
SELECT * FROM Win32_OperatingSystem WHERE ProductType <> '1'


#Applies if the OS is Windows 7 or Server 2008 R2:
SELECT Version FROM Win32_OperatingSystem WHERE Version LIKE
'6.1.%'


#Applies if the computer is a Dell Latitude:
SELECT * FROM Win32_ComputerSystem WHERE Manufacturer LIKE
'%Dell%' AND Model LIKE '%Latitude%'


#Applies if the system has at least 16GB of physical memory:
SELECT * FROM Win32_ComputerSystem WHERE TotalPhysicalMemory >=
16000000000


#Applies if there is at least 500MB available on any drive:
SELECT * FROM Win32_LogicalDisk WHERE FreeSpace > 500000000 AND
Description = 'Local Fixed Disk'


#Applies if the ADMINPAK.MSI software package has been installed:
SELECT * FROM Win32_Product WHERE Name = 'ADMINPAK'


#Applies if located in the eastern time zone:
SELECT * FROM Win32_TimeZone WHERE bias =-300


#Applies if patch KB819696 or KB828026 has been applied:
SELECT * FROM Win32_QuickFixEngineering WHERE HotFixID =
'KB819696' OR HotFixID = 'KB828026'
```

```
#Applies if it is a Monday (0=Sun, 1=Mon, 2=Tue, and so on):
SELECT * FROM Win32_LocalTime WHERE DayOfWeek = 1
```

WMI Security Best Practices (1 of 3)

```
$Creds = Get-Credential
Get-WmiObject -Credential $Creds -Computer "FQDN"
```

- Prefer local administrative accounts over global accounts, assuming all local admin passwords are different, but at least WMI logons are only considered network logons (type 3).
- Require Kerberos with global accounts (-authority).
- Require RPC encryption (-authentication).
- WMI namespace permissions (WMI Control snap-in).
- WMI namespace access logging (WMI Control snap-in).

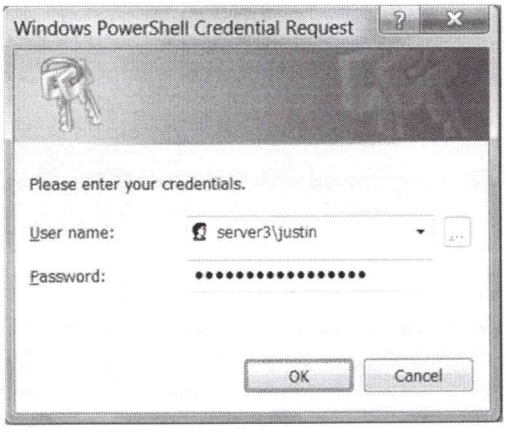
SANS SEC505 | Securing Windows

WMI Security Best Practices (1 of 3)

If you wish to manually authenticate to the remote computer when using WMI, perhaps with a local account at that machine, use the -Credential parameter to specify credentials explicitly.

If you supply a "context\username" string to the -Credential parameter, you'll be prompted with a pop-up dialog box when you run the script to enter the password.

```
get-wmiobject -query "SELECT * FROM Win32_BIOS"
   -computer server3 -credential server3\justin
```



In general, it is better to authenticate with a local account in the Administrators group at the target computer instead of a global user account from Active Directory. If the target computer is already infected with malware, the loss of a local account's credentials is far

less damaging than the loss of a global account's credentials, especially when every computer has a different password for each of its local user accounts.

If you wish to pop-up the dialog box with empty fields to avoid hard-coding the "context\username" string into your script, use the get-credential cmdlet.

```
# C:\SANS\Day6-Servers\WMI\SecureString-Credentials.ps1

$cred = get-credential

get-wmiobject -query "SELECT * FROM Win32_BIOS"
   -computer server3 -credential $cred
```

The "context\username" string is a plaintext property of the credential object ($creds.username), but the password is stored as a "secure string" and cannot be viewed directly. However, the following code shows how to extract the password from the $cred object in plaintext, so it's not really secure from malware already running as you:

```
# C:\SANS\Day6-Servers\WMI\SecureString-Theory.ps1

$cred = get-credential

$bstr = [System.Runtime.InteropServices.Marshal]::`
                        SecureStringToBSTR($cred.password)

[System.Runtime.InteropServices.Marshal]::PtrToStringAuto($bstr)
```

So, don't be fooled by Microsoft's documentation: "secure strings" are not secure in memory against malware or hackers that can inject DLLs into your PowerShell process or which can scrape the raw virtual memory of your process. If you do store a password in a variable, set that variable to $null immediately after authentication, remove the variable by name, and then either terminate the PowerShell process or hope the .NET garbage collector quickly cleans it up.

To encourage the .NET garbage collector to work sooner rather than later:

```
[GC]::Collect(0)
```

Require Kerberos for Global Accounts

For security reasons, it's best to only use Kerberos when authenticating over the network since this helps to block certain relay authentication attacks when using NTLM. And, in general, use Group Policy to only allow NTLMv2, never NTLMv1 or LM.

To authenticate with Kerberos only, use the -Authority parameter with an argument like "kerberos:*domain\hostname*" and the -Computer parameter with the hostname, FQDN or IP address of the target. Note that you cannot use an IP address for the -authority parameter, but you can use an IP address for the -Computer parameter. For

the -Authority parameter, you can use either the NetBIOS name of the domain (sans) or its DNS name (sans.org), but the hostname for the -Authority parameter should be just the simple hostname (server3) and not the FQDN (server3.sans.org).

The following commands all work to compel the use of Kerberos:

```
get-wmiobject -query "SELECT * FROM Win32_BIOS" -authority
   "kerberos:sans.org\server3" -computer server3.sans.org

get-wmiobject -query "SELECT * FROM Win32_BIOS" -authority
   "kerberos:sans\server3" -computer server3

get-wmiobject -query "SELECT * FROM Win32_BIOS" -authority
   "kerberos:sans.org\server3" -computer 10.4.8.2

get-wmiobject -query "SELECT * FROM Win32_BIOS" -authority
   "kerberos:sans\server3" -computer 10.4.8.2
```

Require RPC or WSMAN Encryption

RPC encryption is used by default, but as a precaution you may specify that RPC encryption be mandatory. If you want to ensure that encryption is used for the RPC connection, include the "-Authentication 6" option (the number 6 is for the PacketPrivacy connection security level for RPC).

The following command forces the use of both Kerberos and RPC encryption:

```
get-wmiobject -query "SELECT * FROM Win32_BIOS" -authentication 6
   -authority "kerberos:sans.org\server3"
   -computer server3.sans.org
```

When writing functions or script that use WMI, adding the above extra parameters and arguments is no big deal.

When using the CIM cmdlets, WSMAN is used by default instead of RPC. RPC will only be used if requested (New-CimSessionOption -Protocol Dcom).

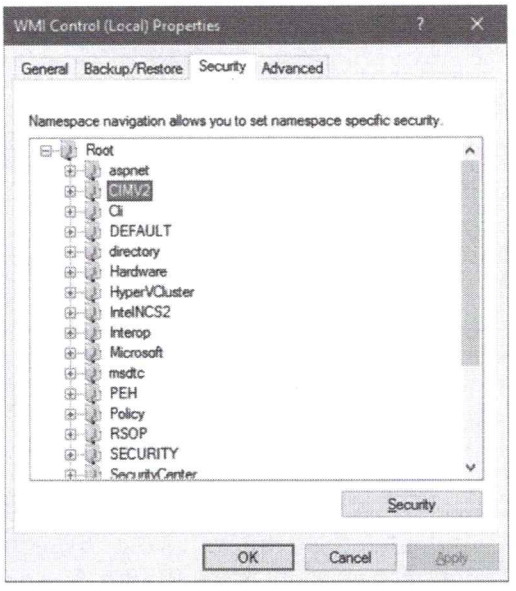
When using the CIM cmdlets, never use the -NoEncryption switch when creating a new WSMAN session with New-CimSessionOption. Inside the LAN, it is fine to use the default encryption, but use SSL/TLS when going over the Internet (-UseSSL switch).

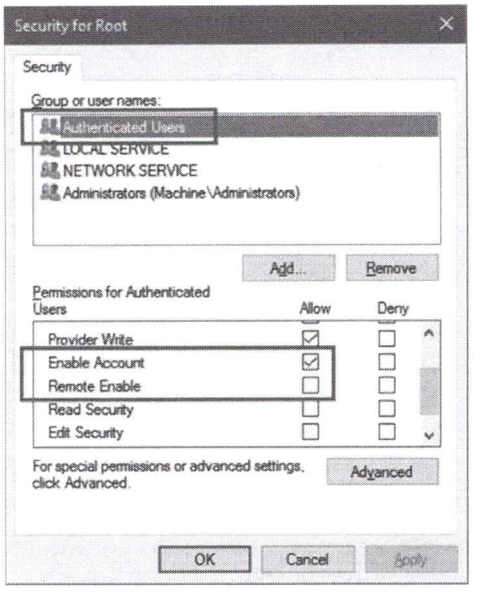
WMI Namespace Permissions

By default, the Authenticated Users group is granted permission to access WMI locally, but not over the network. Only members of the Administrators group are granted remote access permission to the WMI service. But these permissions are not written in stone.

To see your WMI namespace permissions, open an MMC console (mmc.exe) > File menu > Add/Remove Snap-In > add the WMI Control snap-in for the local or remote

computer > OK > right-click the WMI Control snap-in in your MMC console > Properties > Security tab.





When you examine the permissions for the Root namespace, you will see that the Authenticated Users group has the "Enable Account" permission, but not the "Remote Enable" permission (it is unchecked). The "Enable Account" permission is for local access to WMI and the "Remote Enable" permission is for over-the-network access to WMI. The Administrators group has "Remote Enable" by default.

For role-based access control, a new group could be granted remote access to the WMI service, but without granting that group the excessive privileges and permissions of the Administrators group.

WMI Requires The Network Logon Right

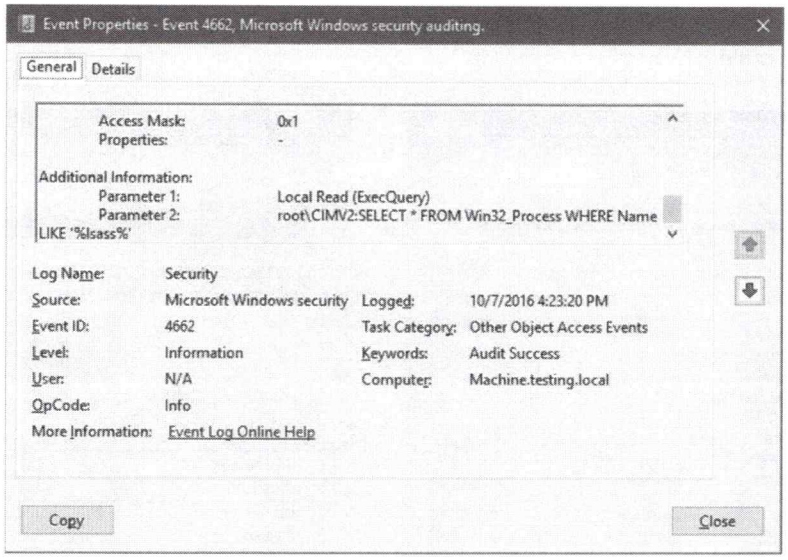
To remotely access the WMI service also requires the "Access to this computer from the network" logon user right. User rights can be managed through Group Policy, INF security templates, and PowerShell Desired State Configuration (DSC) too.

WMI Namespace Access Logging

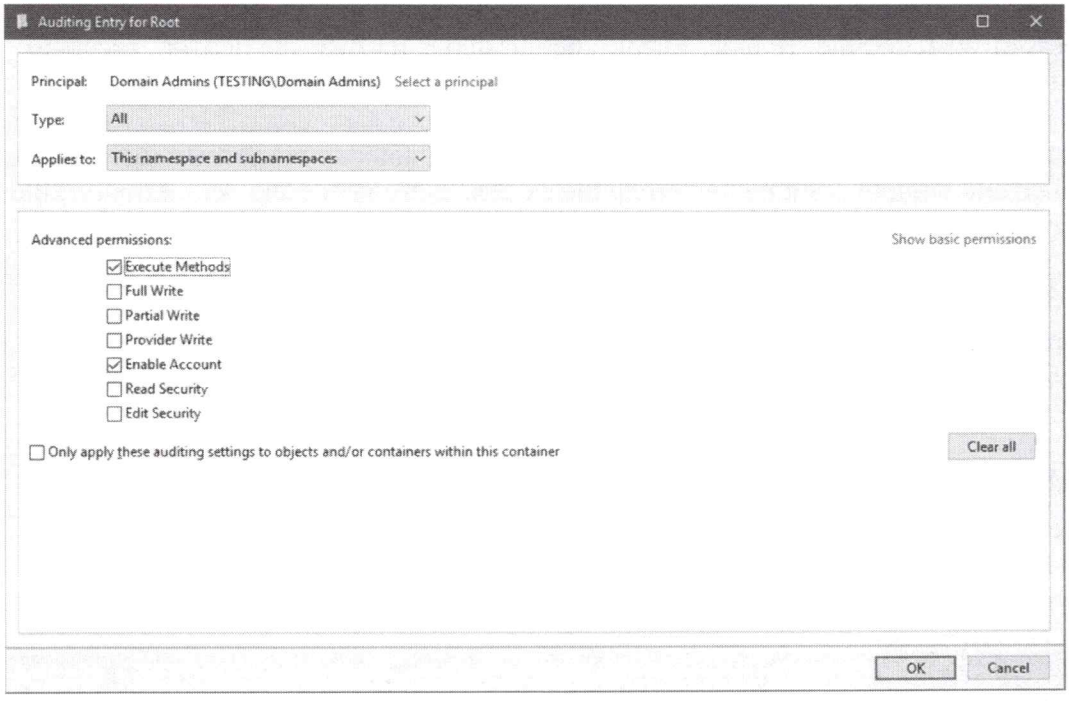
To enable the logging of WMI queries and commands, you must 1) enable the "Other Object Access Events" advanced audit policy subcategory, and 2) configure the audit settings on the namespace(s) desired using the WMI Control snap-in.

```
auditpol.exe /set /subcategory:"Other Object Access Events"
   /success:Enable /failure:Enable
```

This will write event ID 4662 to the Security log showing the user name, WMI namespace, WMI class accessed, and the WMI method invoked, but only for those users whose groups have been added to the audit SACL of the namespace accessed.



No WMI access is logged by default, even with the "Other Object Access Events" audit policy enabled. You must also configure the Auditing tab (also known as the "System Access Control List" or SACL) for the desired namespaces, such as the Root namespace, and choose the group(s) whose access will be logged. There are several types of actions that can be logged, but the most important is "Execute Methods."



To configure WMI namespace SACLs, right-click the WMI Control snap-in in your MMC console > Properties > Security tab > highlight the Root namespace > Security button > Advanced button > Auditing tab > Add button > select a principal, such as the "Domain Admins" group > click the "Show advanced permissions" link on the right >

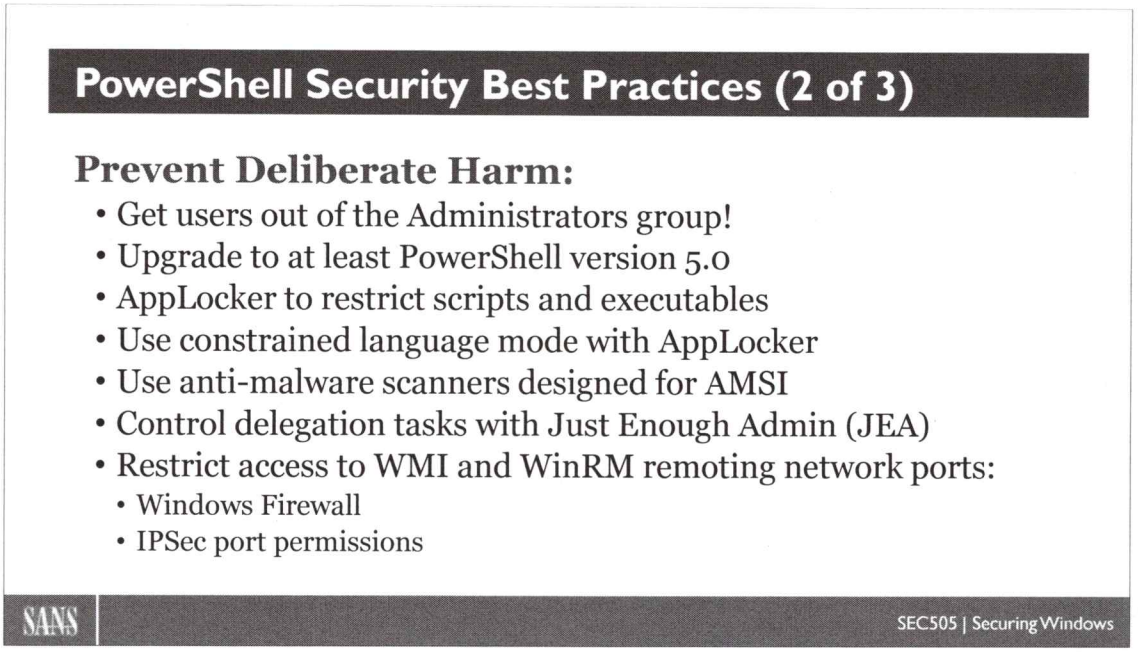
check the boxes for "Execute Methods" and "Enable Account" at a minimum > OK > OK.

Be careful, if you log WMI access from the Everyone group or the local Users group, this will also log access from the local System account. WMI is constantly being accessed by built-in Windows services, so you may accidentally collect a small mountain of data, 99.999% of which would be noise. You might start with logging the "Domain Users" group, then add more groups as necessary. The data will need to be fed into a centralized SIEM for alerting.

WMI Activity Tracing (ETW)

Prior to Windows Vista it was possible to modify the registry in order to write WMI log data to files underneath C:\Windows\System32\wbem\logs; however, for Windows 7 and later operating systems, this feature was discontinued and replaced with WMI support for Event Tracing for Windows (ETW).

Configuring ETW is beyond the scope of this course, the topic is rather complex. For more information, please do a search on the terms "wmi tracing" at the MSDN web site (https://msdn.microsoft.com).



PowerShell Security Best Practices (2 of 3)

From a security perspective, running a PowerShell script is little different than running a compiled binary program. Most hacking tools, including pass-the-hash and DLL injection tools, could be rewritten in PowerShell. Once your adversaries can run PowerShell as Administrator, you have lost control of the box. If a malicious PowerShell script can be run as Domain Admin, every machine joined to the domain could be rooted, sector-scrubbed, infected, or worse.

PowerShell code can also be run entirely in memory without modifying a single bit on the hard drive. The PowerShell engine DLL can be hosted in other processes besides just PowerShell.exe and PowerShell_ISE.exe, including malicious processes. PowerShell code can be invoked from other applications as well, such as Microsoft Office macros, Java standalone apps, VBScript/JScript scripts for the Windows Script Hosts (cscript.exe/wscript.exe), HTA script apps, and CHM help files. And PowerShell has built-in support for executing commands encoded with Base64.

What can be done to help prevent harm when using PowerShell?

We have to make a distinction between 1) deliberate harm by a skilled adversary who does not yet have administrative privileges, and 2) accidental harm caused by non-malicious users. If a best practice helps to prevent deliberate harm, then it also helps to prevent accidental harm. And it would be good 3) to log or audit PowerShell usage as well, preferably with continuous monitoring and alerting.

Get Users Out of the Administrators Group

PowerShell cannot magically circumvent the operating system without an exploitable bug. If a user is not a member of the Administrators local group on his or her computer, then PowerShell scripts cannot be run with administrative privileges either. Getting users out of the Administrators group on their computers is the single most important best practice for securing PowerShell. If an adversary can run PowerShell with administrative privileges, then nothing else matters, all bets are off.

Firewall and IPSec Rules for WMI and WinRM Remoting

Don't forget that Windows Firewall and IPSec rules can restrict remote access to the WMI and Windows Remote Management (WinRM) services to 1) only allow access from the internal IP addresses of IT personnel, 2) to require strong IPSec encryption, and 3) to restrict access to particular global groups of users and/or computers, such as management jump servers.

If adversaries have already stolen the credentials of some administrators, then host-based firewalls are almost the last line of defense for WMI and WinRM.

When using the CIM cmdlets, WSMAN protocol is used by default to talk to the WinRM service to get to the WMI service. WSMAN is the same protocol used for PowerShell remoting too. WSMAN operates on TCP/5985 by default, or on TCP/5986 when using HTTPS with SSL/TLS.

Control PowerShell Remoting

PowerShell remoting allows remote command execution. It is enabled by default on Server 2012 and later, but is not enabled by default on any client operating systems (yet). Because it is so useful, it's best to expect that remoting will not be disabled on servers, and, most likely, will be enabled on clients as well.

PowerShell remoting traffic is encrypted by default using an AES key ultimately protected by the user's password hash, hence, using smart cards or enforcing a good passphrase policy is recommended.

If a certificate is installed, remoting can also be configured to authenticate and wrap the connection in SSL/TLS too. By controlling certificate enrollment, public key size, and revocation checking, then authentication security can be increased. By requiring TLS 1.2 or later with AES 256-bit or better keys, perfect forward secrecy, and SHA-256 or better for hashing, then cipher strength can be maximized.

An alternative to SSL/TLS is to require IPSec for remoting traffic and to limit access to the remoting ports (TCP 5985 and 5986) based on source IP address and the group memberships of the user initiating the connection. Using IPSec instead of SSL/TLS, in fact, may be easier to manage and more secure anyway. Restricting access to the remoting ports in the LAN to just the IP addresses of administrative jump servers and to just a small number of global groups could greatly aid in thwarting PowerShell remoting abuse.

Just Enough Admin (JEA)

By default, only members of the Administrators group at a target machine can connect inbound using PowerShell remoting. But adding someone to the Administrators group just so they can perform a few tasks is far too excessive, it violates the principle of least privilege.

Fortunately, PowerShell remoting rights can be precisely constrained with a set of features called "Just Enough Admin (JEA)" on PowerShell 5.0 and later. With JEA it is possible to 1) grant PowerShell remoting rights to users who are not members of the Administrators group, 2) limit a user to a list of approved cmdlets, 3) limit the arguments which can be passed into these approved cmdlets using regular expression patterns, 4) log the details of every command, and 5) define a run-as mechanism for those cmdlets which require administrative privileges but without exposing any administrative credentials to the constrained remoting user. If you are familiar with *sudo* on Linux, JEA is very similar. And because it is also possible "remote into" one's own local machine, these constraints can be applied locally too.

Application Whitelisting and AppLocker

AppLocker and other whitelisting products can restrict which groups of users are permitted to launch POWERSHELL.EXE and POWERSHELL_ISE.EXE. AppLocker can also restrict exactly which scripts are permitted to execute, assuming that PowerShell is permitted as a process. When possible, restrict PowerShell to the minimum necessary groups.

However, PowerShell is really a DLL (System.Management.Automation.dll) and this DLL can be loaded into more host processes than just POWERSHELL.EXE and POWERSHELL_ISE.EXE. AppLocker and other whitelisting tools need to restrict this DLL, not just the most popular hosting executables.

Constrained Language Mode

With AppLocker and JEA, you can set the PowerShell language mode. The language mode can limit access to certain commands, variables, modules and .NET Framework class types. The more that is limited, the better for security, but the worse for user compatibility. AppLocker and JEA can both set the language mode to "constrained", which helps to prevent PowerShell malware and post-exploitation tools from running.

To read more about PowerShell language modes:

```
get-help about_Language_Modes
```

Anti-Malware Scanners Designed for AMSI

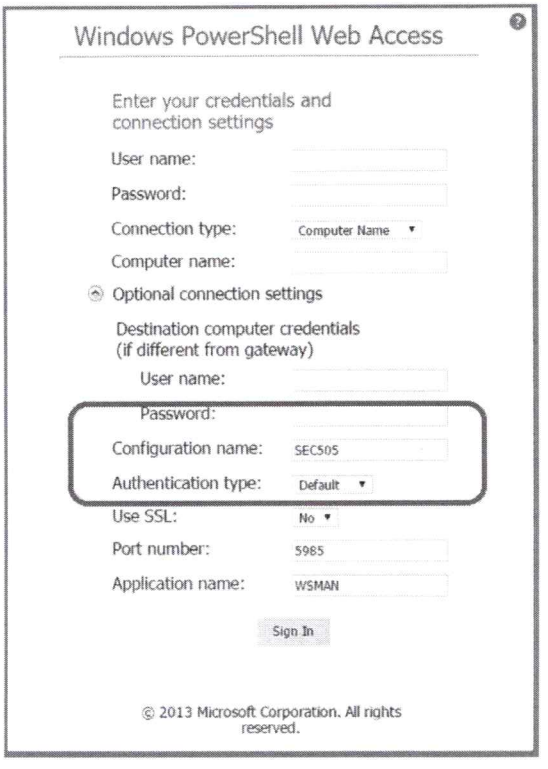
An anti-malware scanner can detect and quarantine malicious PowerShell scripts just like malicious binaries. PowerShell 5.0 and later includes a low-level programming API designed just for anti-malware scanners. The Anti-Malware Scan Interface (AMSI) allows AMSI-capable scanners to examine PowerShell code even when that code is Base64-encoded or otherwise obfuscated. AMSI grants the scanner access to the code

after de-encoding and right before execution. AMSI-capable scanners can also examine code downloaded from the Internet, dynamically generated code, or code which exists only in memory. Even better, AMSI is also for the Windows Script Host executables (wscript.exe and cscript.exe) which run VBScript and JScript files.

Look for the "AMSI" acronym when you next upgrade your anti-malware solution.

PowerShell Web Access On IIS (PSWA)

An optional web application that may be installed on IIS is PowerShell Web Access (PSWA). PSWA allows any JavaScript-capable browser to connect to the PSWA application on the IIS server over SSL/TLS and use PowerShell remoting to execute commands on other machines inside the LAN. The browser could be Safari, for example, on an Apple iPad tablet connecting to https://<*fqdn*>/pswa/. The IIS web server must be Windows Server 2012 or later.



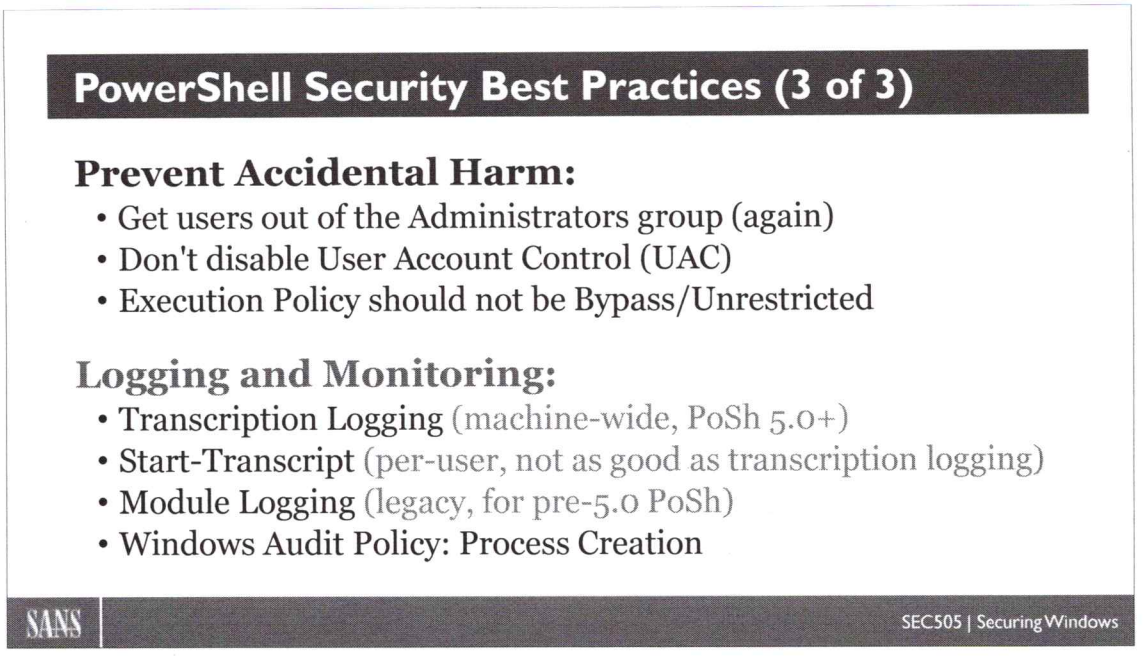
For securing PSWA, here are the most important best practices:

- Do not install PSWA if it is not needed.
- If possible, only allow access to the PSWA IIS server through a VPN gateway that restricts connections to just those groups who require remote access.
- Keep the IIS server fully up-to-date with patches.

- Use the Add-PswaAuthorizationRule cmdlet to tightly control 1) which groups may log into PSWA, 2) which group of computers each user group may remote into inside the LAN, and 3) each group's associated Just Enough Admin (JEA) endpoint configuration. (It is not the case that, once a user logs into PSWA, that that user may remote into any box in the LAN: each user group to computer group mapping must be explicitly added to be allowed. As seen in the PSWA screenshot above, the user can be required to enter the JEA configuration name when connecting with their browser. JEA blocks all commands by default, allows only the commands and arguments granted, and logs commands to transcript files.)

- Disable SSL, require at least TLS 1.2, use a cipher suite which supports large keys (at least 256-bit AES and 2048-bit RSA) and Perfect Forward Secrecy.

For more information about how to install and configure PSWA, do an Internet search of the term "Install-PswaWebApplication."



PowerShell Security Best Practices (3 of 3)

Accidental harm is also mitigated by following the best practices for mitigating deliberate harm. The best practices for deliberate harm should be done first, but there are some extra steps that can be taken to help users avoid accidentally hurting themselves. And if these steps help to deal with hackers too, then great!

User Account Control

Windows Vista and later includes a security feature named "User Account Control (UAC)" which causes programs to be launched by default with limited privileges even if you are logged on as someone who is a member of the local Administrators group. UAC applies to PowerShell too.

If some users cannot be removed from the Administrators group on their computers, then train them to understand the difference between 1) launching PowerShell as a standard user process and 2) right-clicking PowerShell to "Run As Administrator". Whenever possible, PowerShell should be run as a standard user process. This is especially true if the user is a member of Domain Admins or other high-powered administrative groups.

As a tip, create two shortcuts to launch PowerShell on your desktop, include the name "Admin" in the second shortcut and configure it to always run with elevated privileges (properties of the shortcut > Shortcut tab > Advanced button > check "Run As Administrator"). Modify the shortcut or your profile script to change the background color of the shell to serve as a visual reminder when PowerShell is launched with administrative privileges.

Execution Policy

PowerShell execution policy can be managed through Group Policy. This means that different OUs in Active Directory can have different execution policies assigned to them. Grant the least powerful execution policy to each OU which still permits users and IT personnel to get their work done.

Ideally, this would mean setting the execution policy to Restricted or AllSigned, but at least do not set it to Bypass except when necessary. During user security awareness training, show them what a PowerShell warning looks like, how to cancel execution when the warning is unexpected, and how to notify the help desk or security department.

Execution policy can be assigned at various scopes, such as a global machine policy versus the policy for a particular user. There is a precedence ordering when these policy scopes conflict with each other. Run the following commands to read the description of how the precedence ordering works and how they can be managed:

```
get-executionpolicy -list | format-table -autosize

get-help about_Execution_Policies
```

Keep in mind, though, that execution policy is mainly intended to prevent non-malicious users from accidentally shooting themselves in the foot. If a user or attacker is a member of the Administrators group and seeks to undermine security, then the execution policy does not matter. Even when users are not in Administrators, there are still other methods to get around execution policy restrictions, e.g., simply pasting commands from the clipboard into the shell.

Per-User Command History Logging: Start-Transcript

It is possible to log every command a user executes, including the date, time, and command-line parameters. This can be logged to a local file, but then this file could be edited or deleted. This logging information could instead be written over the network, with every command, to a file in an SMB shared folder whose NTFS permissions allow data to be appended, but not edited or deleted. On that SMB server, another process could monitor the command history log files or periodically import them into a security information management system, e.g., Splunk.

For more information about logging user commands:

```
get-help Start-Transcript -full
```

Machine-Wide Transcription Logging (PowerShell 5.0+)

In PowerShell 5.0 and later, command transcription was greatly enhanced to cover all commands, including memory-only code in scriptblocks that have been Base64-encoded. This is very important for incident response, forensics, and the Hunt Team.

To enable machine-wide transcription logging with PowerShell 5.0 and later, see the GPO options named "Turn on PowerShell Transcription" and "Turn on PowerShell Script Block Logging" underneath GPO > Computer Configuration > Policies > Administrative Templates > Windows Components > Windows PowerShell.

This can add a large volume of log data. This option is similar to automatically running Start-Transcript for every session. Transcript text logs also include metadata such as a timestamp, user identity, run-as identity with JEA, computer identity, and the PowerShell hosting process name and PID number.

Transcript logs may be written to a local disk, which is best for performance, but can instead be written over the network to an SMB share, which is best for security. Permissions should allow users Write and ReadAttributes, but remove all permissions granted to the CREATOR OWNER identity.

Below is a sample of what a transcript log looks like (to avoid line wrapping, "..." has been inserted in various places). It's a regular text file, not a compressed EVTX file. In this sample, the user (TestUser47) was using PowerShell remoting with Just Enough Admin (JEA) restrictions, hence, there is a "RunAs User" too:

```
**********************
Windows PowerShell transcript start
Start time: 20170129070736
Username: TESTING\TestUser47
RunAs User: WinRM Virtual Users\WinRM VA_1454072846_TestUser47
Machine: MACHINE (Microsoft Windows NT 10.0.10586.0)
Host Application: C:\WINDOWS\system32\wsmprovhost.exe -Embedding
Process ID: 7516
PSVersion: 5.0.10586.63
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0.10586.63
BuildVersion: 10.0.10586.63
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
**********************
PS>CommandInvocation(Get-Command): "Get-Command"
>> ParameterBinding(Get-Command): name="Name"; value="Out...
>> ParameterBinding(Get-Command): name="CommandType";...
>> ParameterBinding(Get-Command): name="Module"; value=""
>> ParameterBinding(Get-Command): name="ArgumentList"; value=""
>> ParameterBinding(Get-Command): name="ListImported";...
>> ParameterBinding(Get-Command): name="ErrorAction";...
>> ParameterBinding(Get-Command): name="ShowCommandInfo";...
>> CommandInvocation(Measure-Object): "Measure-Object"
>> ParameterBinding(Measure-Object): name="InputObject"; value=""
>> CommandInvocation(Select-Object): "Select-Object"
>> ParameterBinding(Select-Object): name="Property"; value="...
>> ParameterBinding(Select-Object): name="InputObject"; value=""
```

```
>> ParameterBinding(Measure-Object): name="InputObject";...
>> ParameterBinding(Measure-Object): name="InputObject";...
PS>ParameterBinding(Select-Object): name="InputObject";...
```

You can write your own Hunt Team scripts to search transcription logs for keywords or regular expression patterns that indicate abuse, but far better will be real-time host IDS monitoring products which can send alerts, disconnect remoting sessions, and receive regular vendor updates, similar to how anti-virus products receive regular updates. Otherwise, if you plan to write the search patterns yourself, you'll be playing a never-ending game of catch up as attackers invent new IDS evasion techniques.

Just Enough Admin (JEA) Logging

JEA session endpoints can (and should be) configured to perform transcription logging as well. The example transcript above is an example when using JEA.

Module Logging (Legacy Support)

Even prior to PowerShell 5.0, it is possible to log the activities of specified PowerShell modules in great detail, including command-line parameters and output pipeline contents, but it is only for modules. This feature is called "Module Logging".

Module logging is enabled in a GPO under Computer Configuration > Policies > Administrative Templates > Windows Components. When enabled, you list the names of the PowerShell modules whose commands should be logged (event ID number 800). Again, this transcription logging is per-module only and you must specify the name(s) of the module(s) to be monitored beforehand.

Using transcription logging with PowerShell 5.0 or later is much preferred over module logging. Module logging was not originally designed for security monitoring, it was designed for developers doing troubleshooting and debugging.

Process Creation Audit Policy

There is a Windows audit policy named "Process Creation" in the output of AUDITPOL.EXE or named "Audit Process Tracking" in a GPO or security template. When enabled, this policy creates in the Security log event ID 4688 to log the date, time, user, and EXE of any new process, including PowerShell processes. However, this does not include command-line arguments for security reasons, which drastically limits its usefulness for monitoring, and if the logging of command-line arguments is enabled, this has its own security implications. Nonetheless, these events can still be very useful.

Writing Safer Code

Finally, here are a few tips for writing safer, less exploitable code:

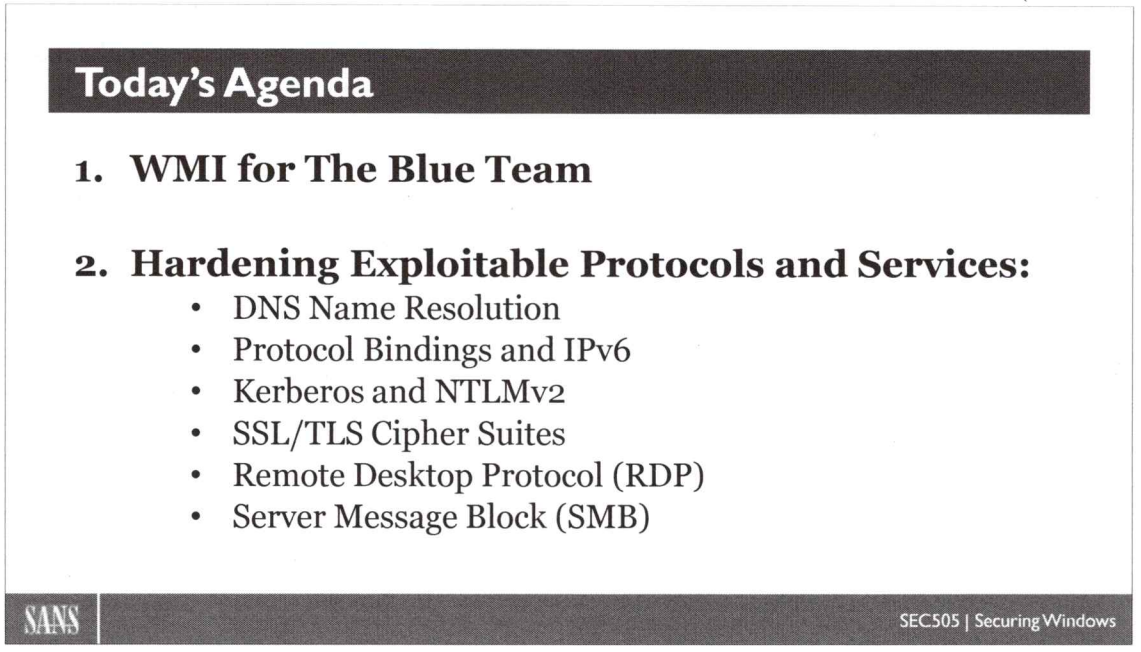
- When you allow untrusted users to execute, but not modify, your scripts, treat all user input as evil until proven otherwise. Such public-use scripts should validate all input to be of the expected type, size and formatting. In general, it is better to filter out all user input by default and only allow what is known to be needed or is

likely harmless. For some applications, filter out dangerous strings known to cause problems, like SQL keywords, EXE names, protected file system paths, etc. JEA can limit the acceptable arguments to a command using regular expression patterns.

- Avoid using the Invoke-Expression cmdlet, and, if you must, be extremely careful about any user input or variables in the strings to be executed.

- Avoid placing passwords, private encryption keys, or other secrets inside scripts. Base64 and other obfuscation tricks will not protect them. It is better to prompt the user for secret input and then convert that data into a "secure string" than to hard-code those secrets into the script (see 'Read-Host -AsSecureString'). You can also use the Data Protection API (DPAPI) to help encrypt data for a particular user on a particular machine (see the Crypto-Examples-Symmetric.ps1 script on the course CD/USB). The Import-CliXml and Export-CliXml cmdlets natively support DPAPI protection of secure strings too. It is safe to include or access plaintext *public* encryption keys, but avoid hard-coding *private* keys or symmetric keys into scripts.

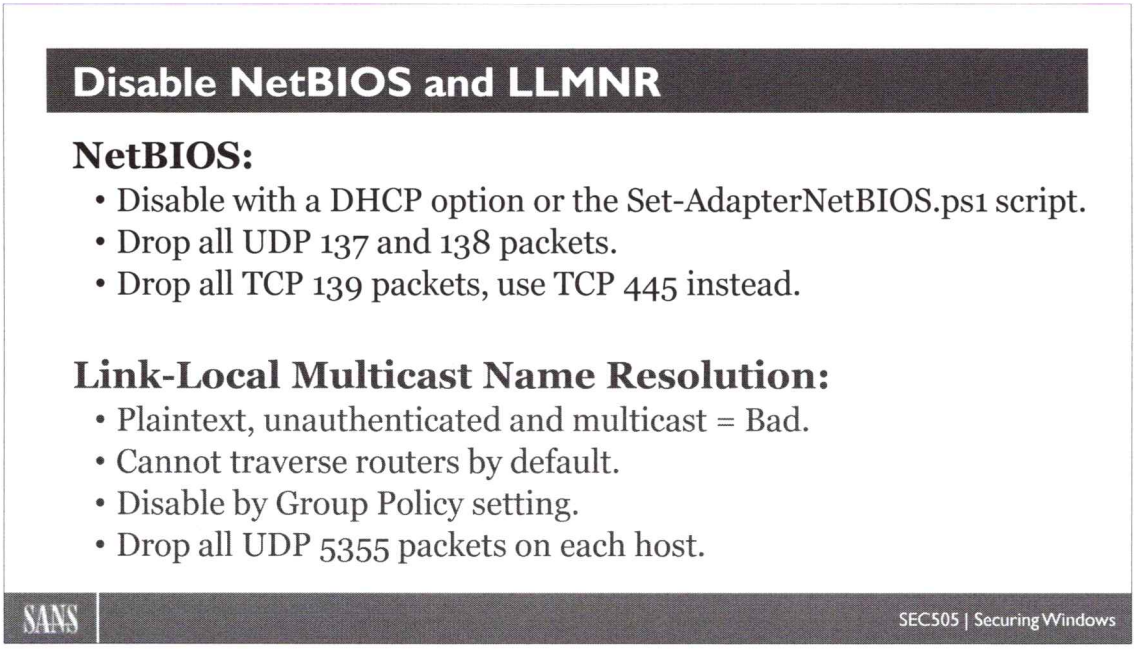
- If possible, write your code so that it is compatible with PowerShell running in "ConstrainedLanguage" language mode (get-help about_Language_Modes).



Today's Agenda

There are several protocols that Windows cannot live without, but these ports and packets are vulnerable to attack. Defensible networking means reducing the exploitability of these protocols as much as practical without disrupting user activities.

In this section, we will talk about adapter bindings, IPv6, SSL/TLS, NTLM, Kerberos, RDP and SMB. In another manual we covered IPSec and the Windows Firewall, which can also be used to secure these other protocols and services.



Disable NetBIOS and LLMNR

NetBIOS is an example of the curse of backwards compatibility. The NetBIOS programming API dates back to 1983. Support was later added for NetBEUI, IPX/SPX and then Microsoft's NetBIOS Over TCP/IP (NetBT) protocol (RFCs 1001 and 1002). When referring to "NetBIOS" today, the term usually refers to the NetBT protocol, not to the programming API. As an API, NetBIOS implements a naming scheme, name registration, name resolution, connection-oriented session management, and connectionless transactions. As the NetBT protocol, NetBIOS operates on TCP 139 for connection-oriented communications, and UDP 137 and 138 for connectionless communications. Instead of broadcast name resolution, a WINS server can be used as a central repository for all NetBIOS names in the organization.

NetBIOS is fading away because it does not scale to large organizations, it consumes a lot of bandwidth, and DNS is the preferred name resolution method on the Internet.

From a security standpoint, NetBIOS is mainly a reconnaissance risk because it is plaintext, uses sniffable broadcasts, and computers can be remotely queried for their NetBIOS names in a way which reveals too much information to unauthenticated attackers. There have also been exploits against the NetBIOS-related ports on Windows, such as SMB when accessed on TCP 139, but new exploits against these are now rare.

However, when SMB runs over TCP 445, NetBIOS is not used. Again, with SMB access to shared folders or printers, and the various RPC-Over-SMB protocols, these protocols do not require NetBIOS in order to function.

To resolve the IP address of a computer (server47) with its NetBIOS name:

```
nbtstat.exe -R              #Clears the local NetBIOS name cache.

nbtstat.exe -a server47     #Displays the target's NetBIOS names.

nbtstat.exe -c              #Displays cached name-to-IP mappings.

nbtstat.exe -r              #Displays usage of WINS or broadcast.
```

Disable NetBIOS Over TCP/IP (NetBT) Manually

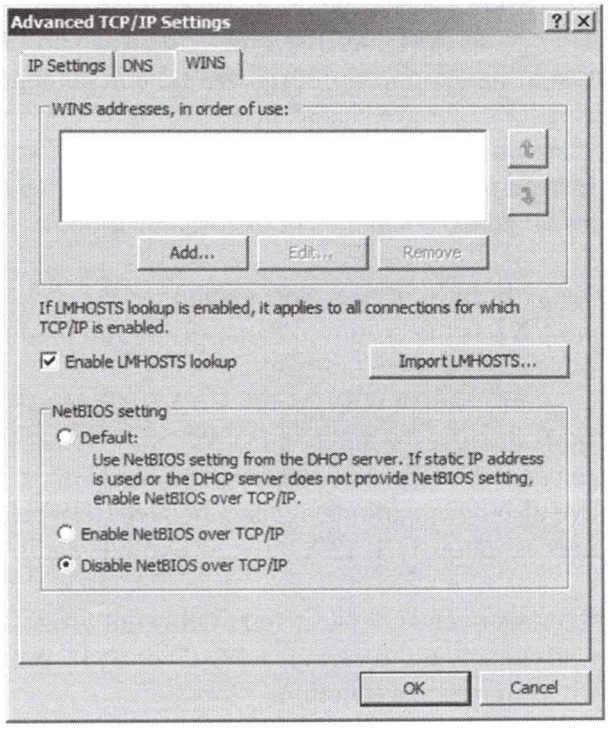
The recommendation is to disable the NetBIOS Over TCP/IP (NetBT) protocol, block all NetBIOS network traffic at the perimeter and on hosts, and retire all WINS servers.

Unfortunately, be prepared to see some very old applications and services fail when you disable NetBIOS, especially ancient network backup programs. Hence, test these changes before configuring it enterprise-wide. If you only run software designed for Windows 2003 or later, then you'll probably be in good shape.

NBT can be disabled manually or through a DHCP scope option.

Try It Now!

To disable NetBIOS Over TCP/IP manually, open Control Panel > Network and Sharing Center > Change adapter settings > right-click the desired NIC > Properties > properties of Internet Protocol Version 4 (TCP/IPv4) > Advanced button > WINS tab > select "Disable NetBIOS Over TCP/IP" > OK.



You should also disable LMHOSTS lookups on this property sheet unless you specifically plan to use that file. A hacker might modify the file with bogus entries.

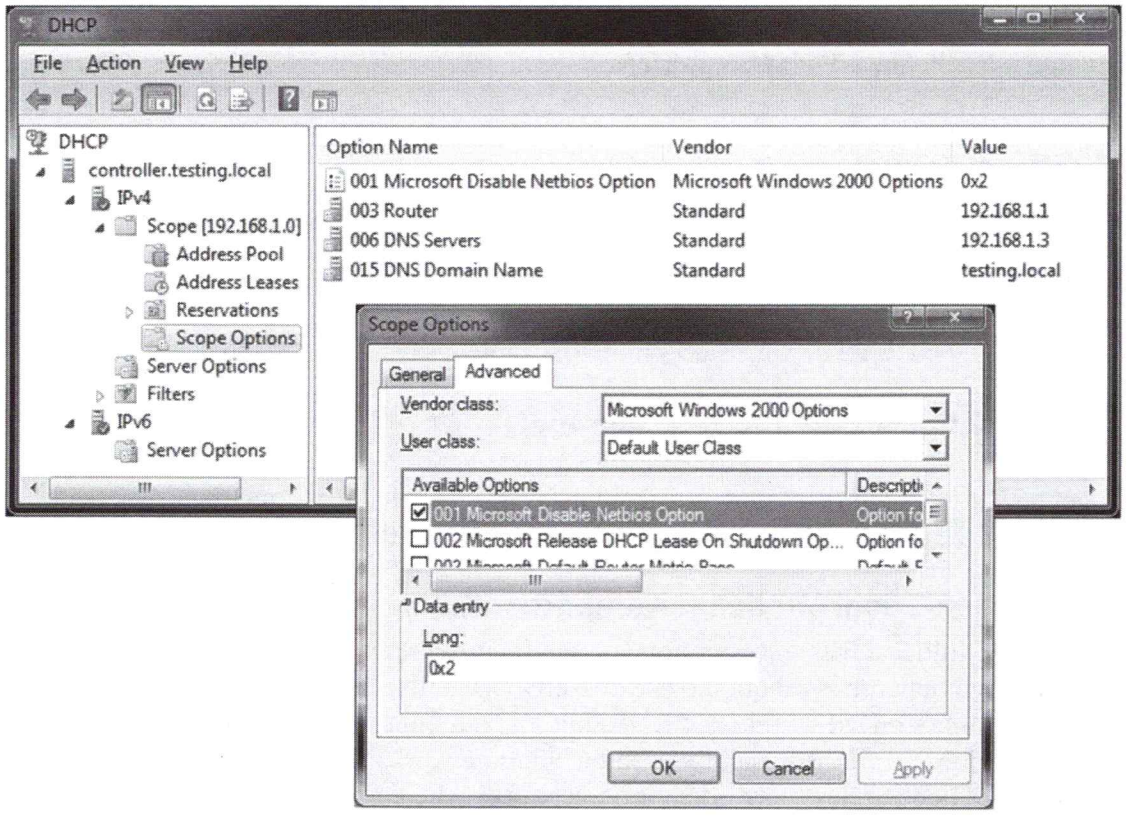
This same option can be set with a script that's on your SEC505 courseware media:

```
.\Set-AdapterNetBIOS.ps1 -NetBiosOption Disable -Verbose
```

This script uses WMI and therefore works against remote computers too. Pipe in a list of computer names to do a mass change. Requires administrative RPC access to each box.

Disable NetBIOS Over TCP/IP (NetBT) Through DHCP

To disable NetBT on large numbers of hosts, use the DHCP scope option named "001 Microsoft Disable Netbios Option", and set it equal to 0x2. This option is found on the Advanced tab when adding a scope option, with Vendor Class set to "Microsoft Windows 2000 Options" and the User Class set to "Default User Class", as seen in the screenshot below.



Drop NetBIOS Packets (Perimeter Firewall and All Host Firewalls)

Additionally, all NetBIOS-related packets should be dropped at the perimeter firewall and on the host-based firewalls of all servers and workstations. This means that the following protocols and ports should be blocked, both inbound and outbound:

- TCP 139 (NetBIOS connection-oriented sessions)
- UDP 137 (NetBIOS name resolution)
- UDP 138 (NetBIOS connectionless datagrams)

In the Windows Firewall, these are part of the "File and Printer Sharing" and "Network Discovery" groups. Make sure to disable the rules for both groups.

However, do not block TCP 445, this is required for SMB. SMB is used for shared folders, shared printers, and RPC Over SMB protocols.

Disable NetBIOS Device Driver (Optional)

If you wish to completely eliminate all NetBIOS support whatsoever, then you must also disable the NETBT.SYS driver. However, there is no DHCP support for this, it's not supported by Microsoft when the change is made on non-servers, and it prevents SMB from connecting out as a client to either TCP port 445 or 139. This is optional, the additional security benefits are very slight once NetBT is disabled programmatically and all perimeter and host-based firewalls are dropping NetBIOS packets. Unless you know that you do not need to act as an SMB client to another server, with or without NetBIOS, then don't disable the NETBT.SYS driver.

To query the start mode and current status of the NETBT.SYS driver:

```
sc.exe qc netbt       # Shows information about the driver.

sc.exe query netbt    # Shows current running status of driver.
```

To set the NETBT.SYS driver's start mode to disabled:

```
sc.exe config netbt start= disabled
```

To get back to defaults, set "start= system" instead. Note that disabling the driver will also prevent the TCP/IP NetBIOS Helper service from starting, which is expected.

What Is Link-Local Multicast Name Resolution (LLMNR)?

Link-Local Multicast Name Resolution (LLMNR) provides name resolution for both IPv4 and IPv6 networks without the use of NetBIOS, WINS or DNS (RFC 4795). It is similar to Apple's mDNS (Multicast DNS) but it is not interoperable with Apple's implementation. LLMNR is mainly used on small networks with no DNS or NetBIOS, such as on ad hoc wireless networks, or on networks where not all hosts have DNS records.

To resolve a hostname (for example, server47) using LLMNR protocol only:

```
Resolve-DnsName -Name server47 -LLMNRonly
```

LLMNR is enabled by default on Server 2008, Vista and later computers. LLMNR is only used with unqualified hostnames lacking periods ("server47") and is not used with fully-qualified domain names ("server47.sans.org") with the exception of the ".local" domain name ("server47.local"), for which LLMNR will strip off the ".local" suffix and then attempt to resolve just the simple hostname by itself ("server47").

When LLMNR is used, a resolver will send a link-scoped multicast request to 224.0.0.242 on UDP port 5355 from an ephemeral source UDP port to the target MAC address of 01:00:5E:00:00:FC with the hostname to be resolved in the payload (or to FF02::1:3 and to 33:33:00:01:00:03 for an IPv6 address). Every suitable host on the link receives the multicast request and examines the requested name. If the host with the desired name is on the link, it responds with a unicast packet back to the sender's IP address and sender's ephemeral UDP port, using UDP port 5355 as the source.

LLMNR packets always have a TTL or Hop Limit set to 1 to prevent passage through routers to other links. Though its RFC allows the use of TCP, Windows just never uses TCP for LLMNR. Windows only uses UDP for LLMNR on port 5355.

How Do DNS, LLMNR and NetBIOS Work Together?

DNS, LLMNR and NetBIOS co-exist peacefully. By default, LLMNR name resolution will be used only after DNS resolution fails for some reason, including the unreachability of any DNS servers. If LLMNR resolution also fails, only then will NetBIOS/WINS resolution be attempted (assuming NetBIOS has not been disabled). So, even if NetBIOS has been disabled, LLMNR can still be used for name resolution when DNS doesn't work. LLMNR is another reason why NetBIOS and WINS are no longer necessary.

To help mitigate cache poisoning attacks, the LLMNR hostname cache is separate from the DNS hostname cache, which is separate from the NetBIOS cache (though it's a bit of a mystery how one views or clears the LLMNR cache). When the three caches are used during resolution, the DNS hostname cache is always consulted first, then the LLMNR cache, and then finally the NetBIOS cache afterwards.

Is LLMNR Secure? Should It Be Disabled?

LLMNR cannot cross routers, which is better for security, but as a plaintext, unauthenticated, multicast protocol, LLMNR is vulnerable to sniffing, MITM and spoofing attacks within the link. Most importantly, though, it's just not needed in well-managed enterprise environments which have DNS dynamic updates enabled.

Hence, with DNS fully deployed and all necessary DNS records correctly updated for the office LAN, it's best to either:

1) disable LLMNR completely, or, as the less-harsh option,

2) use the Name Resolution Policy Table (NRPT) in Windows to control when LLMNR is permitted.

To disable LLMNR completely in Group Policy, go to Computer Configuration > Policies > Administrative Templates > Network > DNS Client, then set the option named "Turn *off* multicast name resolution" to Enabled.

If LLMNR will be completely eliminated, also block UDP 5355 packets, inbound and outbound, at the perimeter firewall and on all host-based firewalls.

On the other hand, to limit LLMNR's use according to the Name Resolution Policy Table (NRPT) in Windows, we'll need to discuss DNSSEC first (which is coming up soon in this manual). This option is less harsh and will allow LLMNR name resolution in limited circumstances when it might be needed.

However, there is one last DNS trick which can be used to make it even less likely that NetBIOS, WINS or LLMNR will be necessary.

DNS Single-Label Name Resolution (The GlobalNames Zone)

There is another shortcoming with both NetBIOS broadcast resolution and LLMNR, they only work to resolve names of other hosts on the same link. This might sound like an argument in favor of WINS, but there is a DNS option to help deal with the problem.

If NetBIOS and LLMNR are both disabled, it is still possible to use DNS to resolve unqualified simple hostnames to IP addresses; for example, it's possible to have "server47" as a CNAME record mapped to "server47.sans.org", which can then be resolved to an IP address normally. It is also possible to do this to maintain uniqueness of these single-label names across all links, across all AD domains, and even across all AD forests in one's organization.

Follow these steps to use DNS for single-label name resolution:

1) Create a new zone in DNS named "GlobalNames" and disable dynamic updates for it. It's preferable to make the zone AD-integrated and replicated forest-wide, but it doesn't necessarily have to be AD-integrated.

2) On every DNS server running Server 2012 or later, open PowerShell and run:

```
Set-DnsServerGlobalNameZone -Enable $True
```

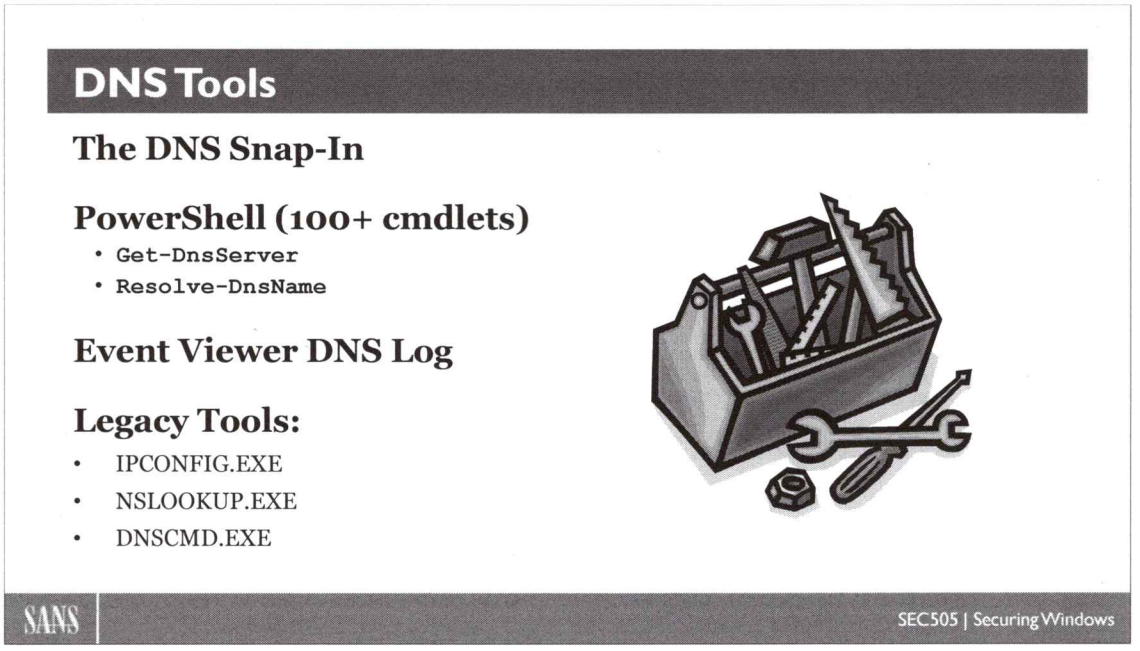
3) On every DNS server running Server 2008 or 2008-R2, run:

```
dnscmd.exe /config /enableglobalnamessupport 1
```

4) Create A, AAAA or CNAME static records in the GlobalNames zone for the unqualified names you want to resolve.

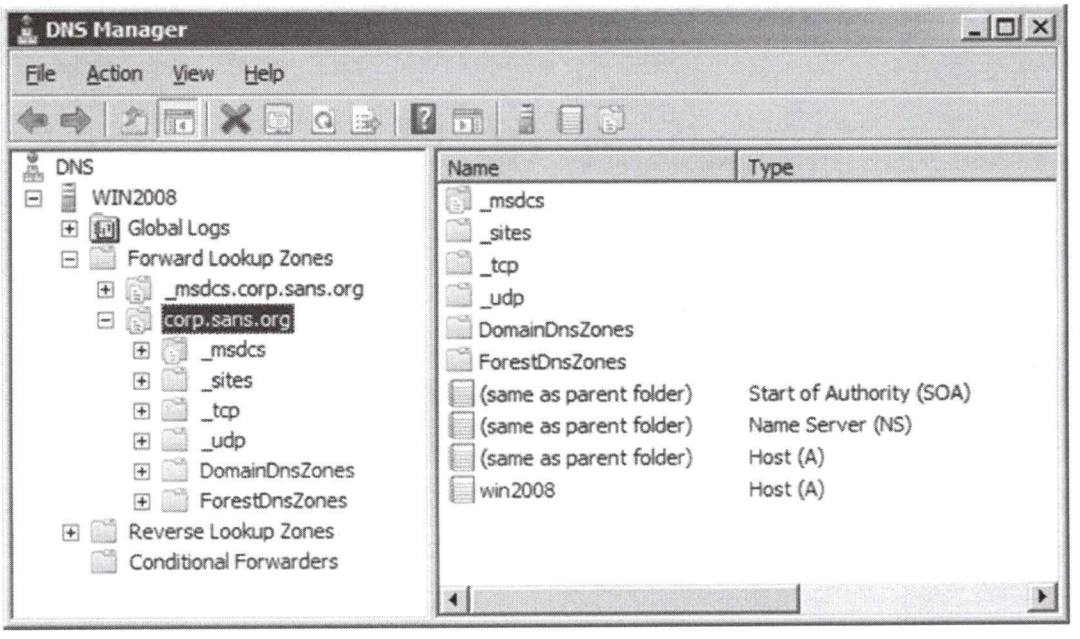
This works because Server 2008 and later DNS servers will attempt to resolve any simple hostnames by looking them up in GlobalNames.

The main limitation of this approach is that every record in GlobalNames must be created, edited and deleted by hand (they're all static), because dynamic DNS updates are not supported. In fact, if a new computer is given a FQDN whose hostname portion matches a hostname in GlobalNames, that computer will not be permitted to use dynamic updates at all. The intention is to keep all the names in GlobalNames unique and to prevent duplicate names anywhere.



DNS Tools

Your primary graphical tool for managing DNS is the MMC snap-in named "DNS". You can find it in the Administrative Tools folder off the Start menu, or, in Server Manager, pull down the Tools menu and select DNS.



PowerShell

There are over a hundred PowerShell cmdlets for managing and querying DNS servers:

```
Get-Command -Module DnsServer

Get-Command -Module DnsClient
```

To show the zones and global configuration settings of a local or remote DNS server:

```
Get-DnsServer

Get-DnsServer -ComputerName mydnsserver.testing.local
```

To add a standard (A) host record to a remote DNS server:

```
Add-DnsServerResourceRecord -A -ZoneName testing.local
   -Name TestHost -IPv4Address 10.9.8.7
   -ComputerName mydnsserver.testing.local
```

To resolve names through DNS of various record types (default is an A record):

```
Resolve-DnsName -Name www.sans.org

Resolve-DnsName -Name sans.org -Type MX

Resolve-DnsName -Name sans.org -Type NS
```

IPCONFIG.EXE

IPCONFIG.EXE is built into Windows and can be used to display IP configuration data, release/renew DHCP leases, display/clear the DNS cache on the client, refresh all dynamic update records in DNS, and more.

NSLOOKUP.EXE

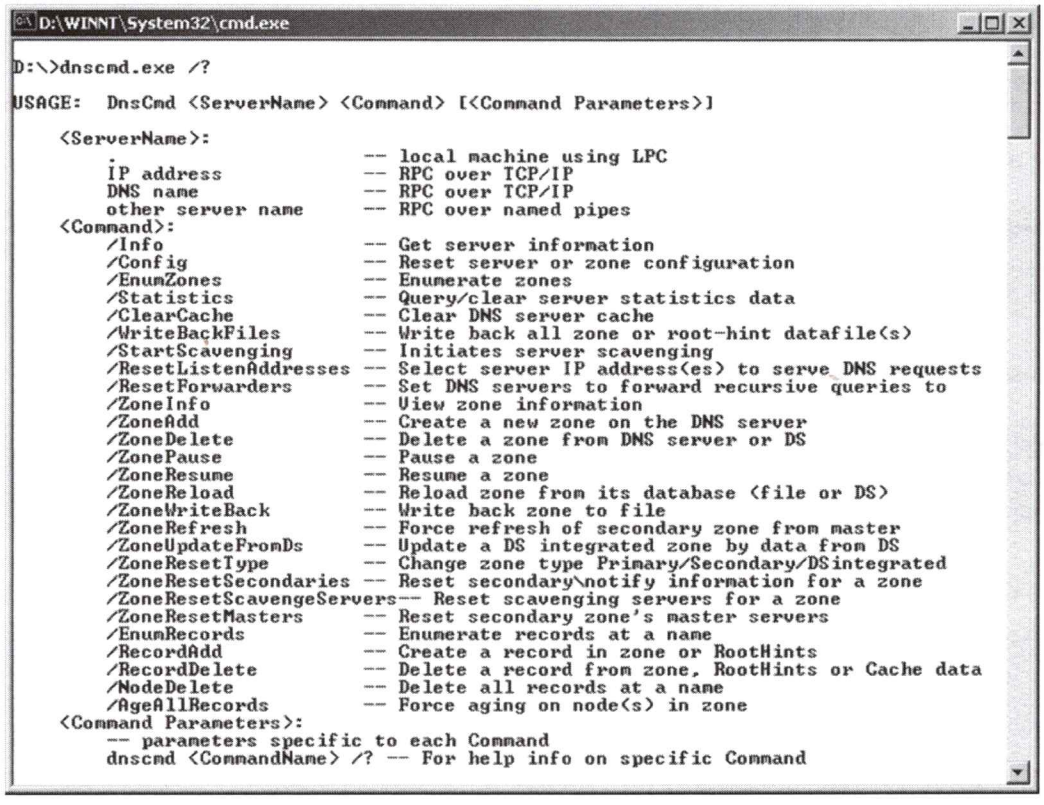
NSLOOKUP.EXE is a command-line resolver. It can be used to issue queries to selected DNS servers to test the information returned. In particular, use the "ls" command to test whether your DNS servers permit zone transfers.

DIG.EXE

DIG.EXE is a popular replacement for NSLOOKUP because it is more versatile. DIG is not a Microsoft tool, it is downloaded for free from the BIND website at http://www.isc.org/products/BIND/. Go to the current release section and get the Windows Binary Kit to get a Windows version of BIND (with DIG.EXE in it).

DNSCMD.EXE

DNSCMD.EXE can be used to manage virtually every aspect of local or remote DNS servers. It can list/create/delete resource records, clear the DNS cache, set IP addresses for recursive queries, sign DNSSEC zones, create/delete zones, initiate zone transfers to secondaries, change zone type, initiate scavenging of stale records, etc.



DNSLINT.EXE

DNSLINT.EXE can perform a variety of DNS-related tests and produce a summary report in HTML format (KB321045). In particular, the tool can verify that the necessary SRV records for Active Directory exist and that DNS delegation is working properly. It can also test DNS, POP3, SMTP and IMAP servers as specified in a configuration text file.

DNSDIAG.EXE

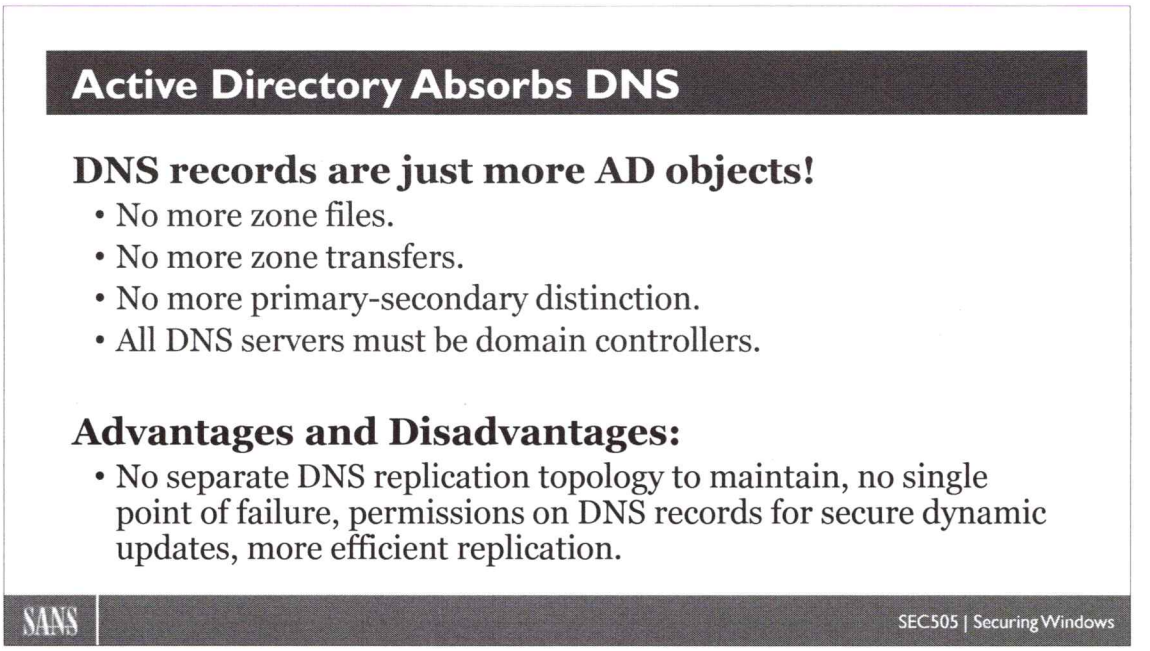
DNSDIAG.EXE is a Resource Kit tool, it's not installed by default. The tool's purpose is to run through the same DNS queries that your e-mail server uses in order to deliver SMTP mail. Use this tool when SMTP gateways are failing because of DNS misconfiguration issues.

Event Viewer

On any machine hosting the DNS service, there is an Event Viewer log just for DNS for the sake of auditing and troubleshooting.

Performance Monitor

Performance Monitor has over 60 counters for charting and logging DNS activity, including counters for queries, zone transfers, dynamic updates, etc.



Active Directory Absorbs DNS

DNS zone data can be stored either in text files or in Active Directory. When stored in AD, each record is stored and replicated separately as an AD object-- the entire zone file itself is not stored as such.

When zone records are stored in AD there are a number of benefits:

- **No Separate DNS Replication Topology:** An independent DNS replication topology does not have to be implemented. DNS data is piggy-backed on top of multi-master AD replication. This simplifies planning and troubleshooting.

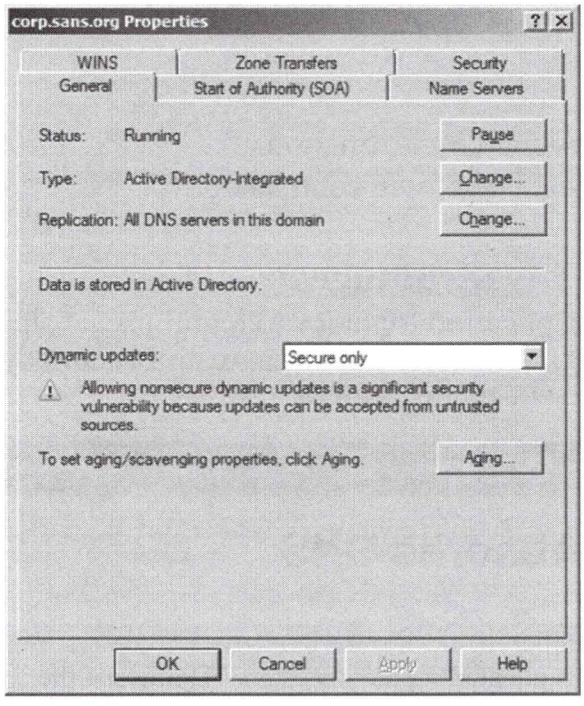
- **No Single Point of Failure:** Because there is no single DNS primary --any server can receive updates-- there is no single point of failure.

- **Permissions on DNS Records:** Each DNS record is an object in AD, and as such each record can have its own permissions, owner and audit settings.

- **Secure Dynamic Updates:** Along with Kerberos authentication, permissions on DNS records can be used to implement a system for secure dynamic updates.

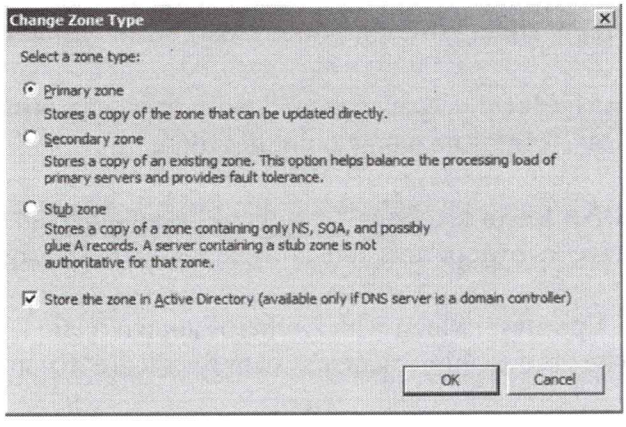
- **More Efficient Replication:** Only the changed properties of individual DNS records are replicated, not entire zone files. And domain controllers are designed to multi-master replicate vast quantities of data in a reliable and speedy way. This makes replication of large numbers of DNS record changes more efficient.

An AD-integrated DNS server can still act as a primary DNS server to non-AD-integrated DNS servers, Microsoft's or otherwise. An AD-integrated DNS server cannot be a secondary.

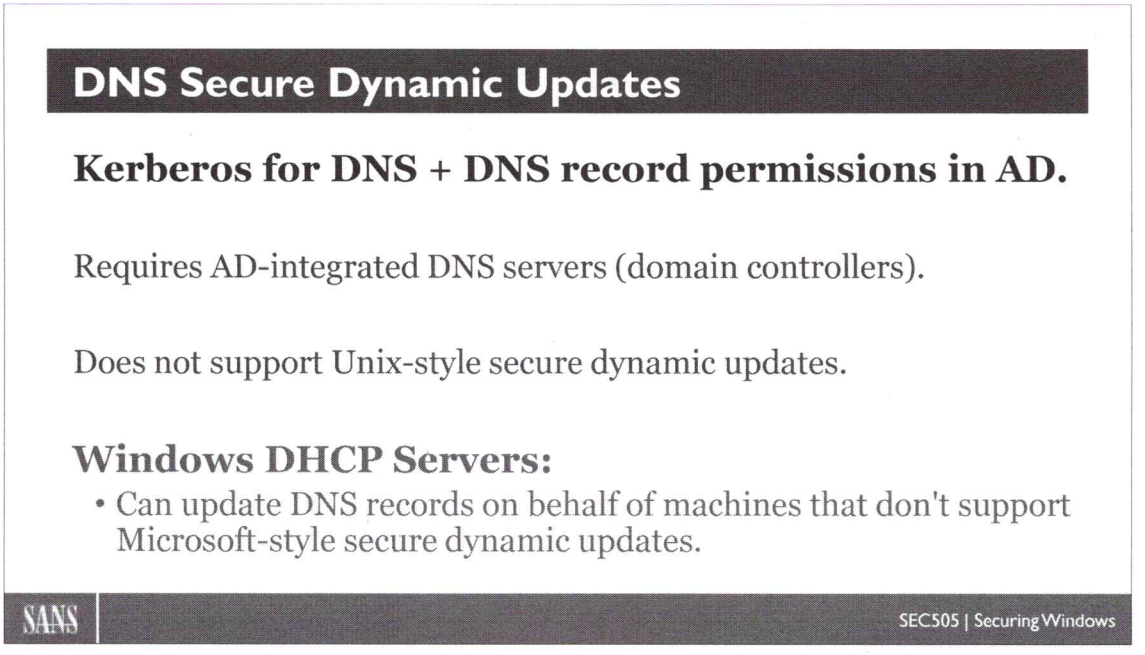


Try It Now!

To store a DNS domain's data in AD, right-click on that domain in the DNS snap-in > Properties > General tab > middle Change button > select Primary Zone > check the box labeled "Store the zone in Active Directory" > OK.



Tip: To import or change a large number of DNS records, switch an AD-integrated zone to become a standard primary. This will cause a standard zone text file to be created. Modify this file as desired, then switch the zone back to AD-integrated. This will import all the records from the zone file into AD.



DNS Secure Dynamic Updates

Windows supports dynamic updates (RFC 2136) and secure dynamic updates (IETF Draft "GSS Algorithm for TSIG" and RFC 2078, but Microsoft secure updates do not follow RFCs 2535 or 2137). Dynamic updates permit DNS records to be changed automatically when a client's IP address changes, instead of requiring an administrator to change the client's records by hand. Secure update requires Kerberos authentication. Hence, clients or DHCP servers which do not support Kerberos will not be able to use secure updates.

Windows systems can update their own address (A) records, and rely upon DHCP servers to update their reverse lookup (PTR) records. Legacy systems rely upon their DHCP servers to update all their records for them.

A primary DNS server supports dynamic updates, but secure updates are possible only with AD-integrated DNS servers. This is because these updates are secured with the assistance of the permissions set on the records themselves in AD. The default permissions allow any authenticated user to create a new record; once created, that user, and only that user, will have full control over the record (administrators still have control of course). If two systems have the same FQDN, the second one to attempt the secure dynamic update will fail.

> **Note:** Make sure the clocks between the client, DHCP server and DNS server are synchronized to within 5 minutes of each other for the sake of Kerberos.

Permissions On DNS Records

When DNS records are stored in AD, a separate access control list of permissions can be applied to each record individually. Permissions are generally left at their defaults for the sake of secure dynamic updates, which rely on these permissions for authorization.

To see DNS records in AD and edit their permissions, use either "AD Users and Computers" or "ADSI Edit" in the /System/MicrosoftDNS container. Before the records can be seen in AD Users and Computers, though, you must enable the viewing of "Advanced Features".

> ***Try It Now!***
> To edit the permissions on a DNS record in AD, open the "DNS Manager" snap-in > expand your DNS server > forward lookup zones > your domain > right-click a DNS record > Properties > Security tab.

Keep in mind that any DNS record with Read access for the Everyone group will be visible to anonymous hackers on the Internet. Write permission is necessary to update the record. With secure dynamic updates, a domain member has the permission to add and modify its own record, but it cannot change the records of other machines. Administrative users, of course, have permission to change any DNS record they wish.

How To Require Secure Updates

Secure updates can be required by the Windows DNS server on a per-zone basis.

> ***Try It Now!***
> To require secure updates on a domain, right-click on that domain in the DNS snap-in > Properties > General tab > Dynamic Updates pull-down menu > select Only Secure Updates.

If secure updates are not required, then any client can change any record. This is because non-secure updates are always attempted first, even if the client is capable of secure updates. If your DNS servers are accessible from the Internet, then hackers can change your mission-critical records to redirect clients to the wrong mail servers, web servers, DCs, etc. Because of the centrality of DNS, corrupted DNS records can seriously disrupt network services.

Windows DHCP

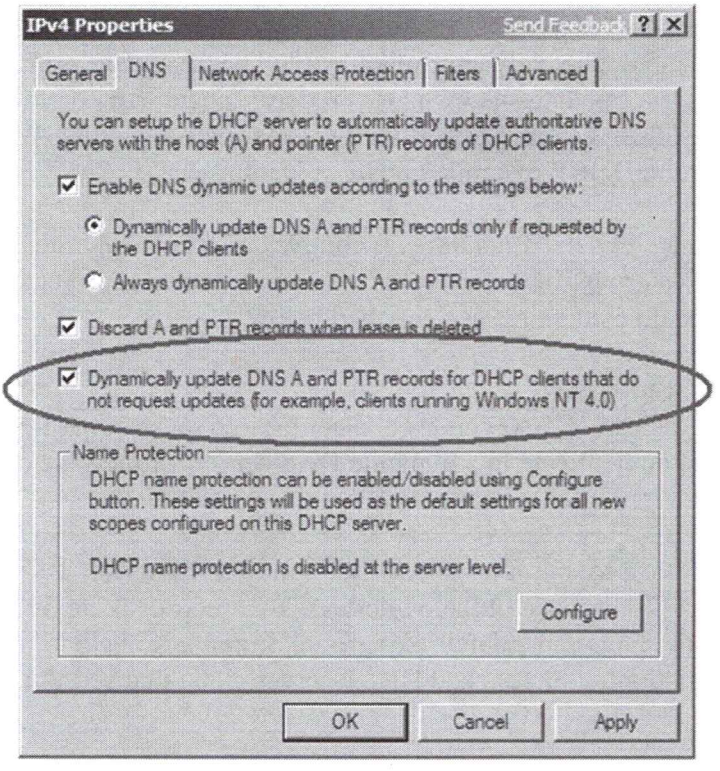
A Windows DHCP server can update DNS records on behalf of clients who cannot do so for themselves. Indeed, even Windows clients rely upon their DHCP servers to update their reverse lookup (PTR) records for them, even though these clients can update their own address (A) records with their DNS servers directly themselves.

> **Note:** Please do not install the DHCP role, the following is an example only.

> ***Try It Now!***

To configure your DHCP server to perform updates for clients, if you have the DHCP role installed, right-click on IPv4 in the DHCP snap-in > Properties > DNS tab > checkboxes labeled "Enable DNS dynamic updates according to the settings below" and "Dynamically update DNS records for DHCP clients that do not request updates (for examples, clients running Windows NT 4.0).

The following screenshot is from the DHCP snap-in, not the DNS snap-in.



Tip: In a Unix BIND environment, have the Windows DHCP server always update DNS records on behalf of DHCP clients, then configure the named.conf file on the DNS server to *allow-update* only from the DHCP server itself.

DNSUpdateProxy Group

If a DHCP server creates/updates DNS records for a client, then the DHCP server owns that record in AD: only that one particular DHCP server can change it. This can be inconvenient if the DHCP server becomes unavailable or if the client is upgraded to Windows (in which case the client should become the new owner).

However, if the DHCP server is added to the DNSUpdateProxy global group, then the DNS records it creates will have no owner. The first client to modify or update the record becomes its owner and will have exclusive control over it like other dynamic records. (Add the computer to the group with the "AD Users and Computers" tool just as you would add a user to a group. You are adding the System account of the DHCP server to the group by doing this.) DNSUpdateProxy is a built-in group which has this

capability designed into the operating system; there is no user right named "Create Ownerless Object" or anything similar to be configured.

> **Important:** If the DHCP server is a member of the DNSUpdateProxy group, then the DNS records it creates are not secure. Anyone who updates or modifies these records becomes their owner with exclusive control over them. This includes the DNS records of the DHCP server itself!

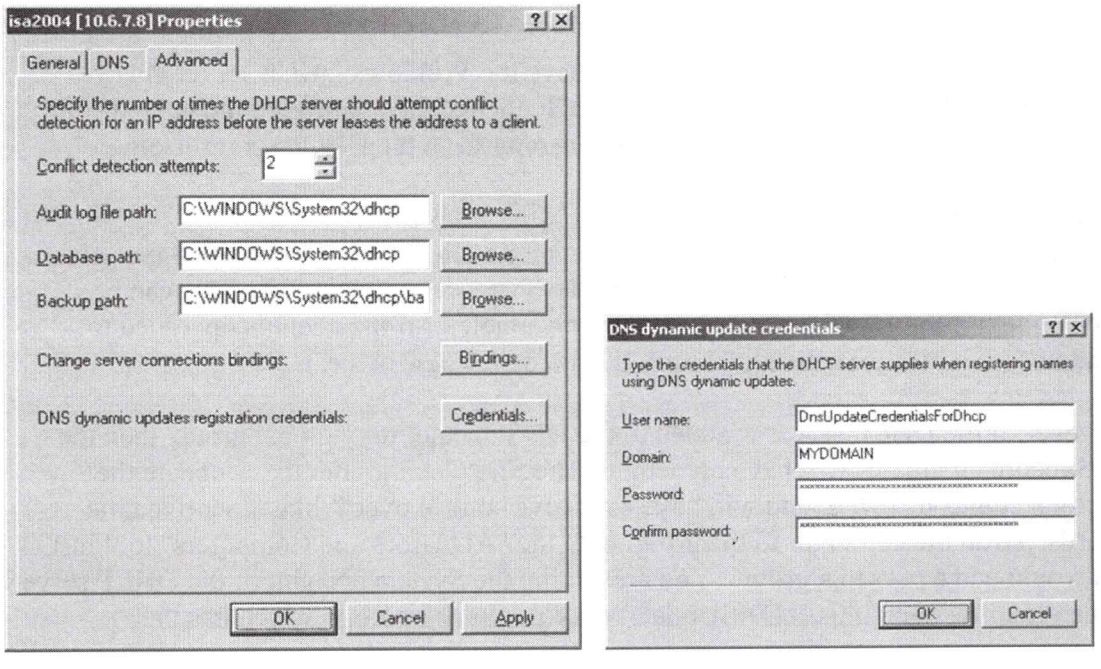
Since the DHCP server updates its own DNS records, then the DHCP server's DNS records may not be owned and exclusively controlled by the DHCP server itself. Hence, after the Windows DHCP server updates its own DNS records, change the permissions on its DNS records to give it exclusive control, or, if the DHCP server is running Windows Server 2003 or later, configure DHCP dynamic update credentials.

> **Warning!** If you install DHCP on a domain controller and you add the server to the DNSUpdateProxy group, then that DC's DNS records are not protected unless you set dynamic update credentials as described below.

DHCP Dynamic Update Credentials

Windows Server 2003 and later DHCP servers can be configured to use a hard-coded set of credentials with which to authenticate to a dynamic DNS server for the sake of secure dynamic updates on behalf of the clients obtaining IP addresses from it.

This feature should always be used, but it is especially important if the DHCP service is installed on a domain controller because of the ownerless DNS records issue discussed in the section above concerning the DNSUpdateProxy group. It may also help to limit the damage caused by a malfunctioning or attacked DHCP server when that server is a domain controller.



Try It Now!
On a Windows Server 2003 or later system running the DHCP service, create a global user account named "DnsUpdateCredentialsForDhcp" and add it to the DNSUpdateProxy group. Next, open the DHCP snap-in on the DHCP server > right-click the server > Properties tab > Advanced tab > Credentials button > set the username, domain and passphrase > OK > OK.

The same credentials can be used on all DHCP servers in the forest. The account used should be added to the DNSUpdateProxy group, but it should not be added to any administrative groups. Because the account will not be a member of any administrative groups, it's OK to give it an obvious/descriptive name. However, because the account is likely to be forgotten, give it a passphrase of 20+ characters and set its passphrase to never expire.

DHCP Audit Logging
It's important to enable audit logging in DHCP so that link-layer MAC addresses can be mapped to IP addresses (and ultimately to computers and people) for the sake of incident response and forensics. To enable logging in the DHCP snap-in, right-click the IPv4/IPv6 container > Properties > General tab > check the box "Enable DHCP audit logging". To change the default folder where the logs are stored, see the Advanced tab on that same property sheet.

Miscellaneous Registry Values For DNS

There are a few registry values related to hostname resolution that should be mentioned.

QueryIpMatching

Hive: HKEY_LOCAL_MACHINE
Key: \System\CurrentControlSet\Services\DnsCache\Parameters
Value Name: QueryIpMatching
Data Type: REG_DWORD
Data: 0 or 1 (default is 0)

When QueryIpMatching is set to 1 on a system, the system will only accept DNS resolution replies from the same IP address of the DNS server originally queried. Hence, if a computer queries a DNS server at 10.0.0.1, then the client will only accept query responses from 10.0.0.1. Setting this value will help to prevent the poisoning of the hostname cache on the computer. This value is not related to "DNS cache poisoning" on DNS servers however.

DisableDynamicUpdate

Hive: HKEY_LOCAL_MACHINE
Key: \System\CurrentControlSet\Services\Afd\Parameters
Value Name: DisableDynamicUpdate
Data Type: REG_DWORD
Data: 0 or 1 (default is 0)

This registry value can be used to disable dynamic updates entirely on a client or server, preventing that machine from attempting to update its DNS records.

RegisterDnsARecords

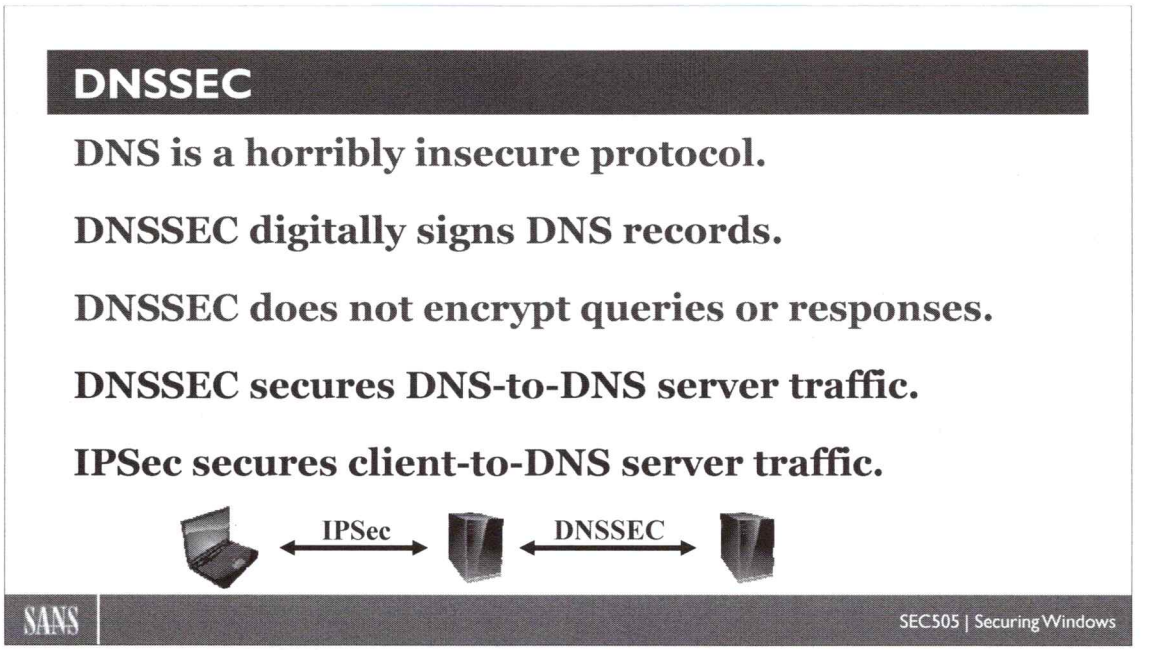
Hive: HKEY_LOCAL_MACHINE
Key: \System\CurrentControlSet\Services\Netlogon\Parameters
Value Name: RegisterDnsARecords
Data Type: REG_DWORD
Data: 1 = register (default), 0 = do not register

The RegisterDnsARecords values determines whether a domain controller should attempt to dynamically register its records related to being a domain controller (all the records found in the Netlogon.dns file). This value only relates to DCs. Disabling dynamic updates entirely also disables these.

UpdateSecurityLevel

Hive: HKEY_LOCAL_MACHINE
Key: \System\CurrentControlSet\Services\Afd\Parameters
Value Name: UpdateSecurityLevel
Data Type: REG_DWORD
Data: 0x00000000 (default), 0x00000010, or, 0x00000100

This value controls how clients attempt to update their own records on their DNS servers. The default (0 in decimal) configures the client to attempt insecure update first, then secure update if necessary. Setting the value to 0x00000010 (16 in decimal) will cause the client to use only insecure updates. Setting the value to 0x00000100 (256 in decimal) will cause the client to use only secure updates.



DNSSEC

DNS is critical to network operations. But standard DNS is insecure because it is unsigned, unencrypted, normally uses UDP instead of TCP, there is no authentication of the DNS server before a client relies on it, and there is typically no authentication of clients before the server responds to their requests. Hackers can sniff DNS packets to extract information, modify and retransmit intercepted packets without detection, spoof DNS replies, and poison the caches on both DNS servers and clients.

To deal with *some* of the above threats, we can augment standard DNS with Domain Name System Security Extensions (DNSSEC). The purpose of DNSSEC is to authenticate the source of DNS replies as being authoritative, provide integrity verification of replies, and provide proof of the non-existence of a DNS record.

Notice that DNSSEC does not encrypt requests or replies, does not secure zone transfer channels, does not authenticate clients, and will not help to prevent DoS attacks against DNS servers. Indeed, DoS attacks might be slightly easier with DNSSEC. Because of these shortcomings, we'll see that IPSec is still recommended for some DNS traffic as well. DNSSEC is not a panacea, but at least it is backwards compatible with clients which do not use or understand DNSSEC records.

Requirements

Strictly speaking, DNSSEC is supported on Server 2008-R2, but, as a practical matter, plan to only use Server 2012 and later. Server 2008-R2 DNSSEC is not compatible with AD-integrated zones of authority or with dynamic updates, and the management of keys is much more difficult than on Server 2012 and later. On Server 2012 and later, DNSSEC is compatible with primary zones, secondary zones, AD-integrated zones,

secure dynamic updates, zone transfers, forwarding, and the mixture of Microsoft and non-Microsoft DNS servers. When joined to a domain, Server 2012 and later can also distribute trust anchor keys automatically to other domain-joined DNS servers throughout the forest. Hence, only use Server 2012 and later for DNS servers with which you intend to use DNSSEC.

On the client side, only Windows 7, Server 2008-R2 and later operating systems have any DNSSEC-related features. Windows XP, Windows Vista and Server 2003 have none.

Firewalls do not need to be modified. DNSSEC uses the same ports as regular DNS, namely, both UDP and TCP port 53. For very old firewalls, ensure that UDP 53 packets with more than 512 bytes of data are permitted through (this is for EDNS0).

Clocks on DNS servers must be kept synchronized and accurate within five minutes.

If you wish for other DNS servers around the world to validate your records, your domain name registrar must support DNSSEC too. More specifically, your registrar must allow you to manage the DS and/or DNSKEY records for your domain (most do).

If your DNS server forwards all of its queries recursively to another DNS server, perhaps at your ISP, first test whether or not that other DNS server will agree to resolve, cache and respond with any DNSSEC-related records at all. Shamefully, some public DNS servers will simply drop or ignore all DNSSEC queries! In this situation, DNSSEC validation will fail for any zones for which you have configured a trust anchor key on your DNS server; you can contact the administrator of the DNS server to which you are forwarding your queries (good luck) or switch to a different DNS server for forwarding (faster and less stressful).

Key DNSSEC Concepts

Here are a few key concepts and terms for understanding DNSSEC. Fortunately, it is not necessary to know every detail of DNSSEC before being able to deploy it (feel free to read RFC 4033 to get started). The important concepts will be illustrated with the tools shortly.

DNSSEC uses new record types, including DS, DNSKEY, RRSIG and NSEC3 records. These are visible in the DNS manager tool like any other record.

A DNS server's response to a client (to an endpoint "stub resolver" or to a DNS server acting as a client) may include the requested resource records plus associated RRSIG records, which are the digital signatures of the requested records. The signature was created with a private key on the DNS server.

Desired Record ← RRSIG

[Where RRISG = Digital signature of the desired DNS record.]

"Zone signing" is the process of creating RRSIG signature records for the other records in a DNS zone. This zone may be AD-integrated and use dynamic updates. When records are added, modified or deleted, the entire zone does not have to be resigned, only the new records are signed.

To validate the RRSIG signature for the requested DNS record, the validating client obtains the DNS server's corresponding public key by requesting the appropriate DNSKEY record. This "client" is usually another DNS server, not a tablet or laptop. The DNSKEY record contains the public key which corresponds to the private key used to create the RRSIG record, i.e., the signature of the original DNS record requested by the client.

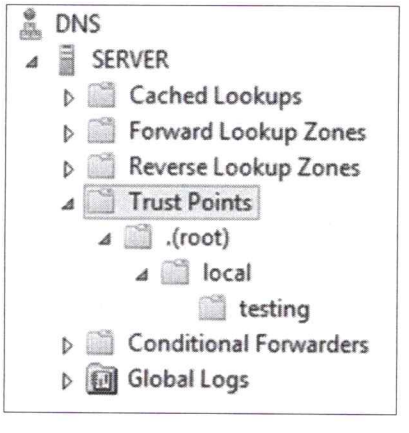
Desired Record ← RRSIG ← DNSKEY

[Where DNSKEY = Public key needed to validate the RRSIG signature record.]

Trust Anchor

A "trust anchor" is a public key (or the hash of a public key) which is contained in a special DNSKEY record (or a DS record, when it just contains the hash). DNS domains and zones of authority exist in a hierarchical parent-child tree. The trust anchor DNSKEY public key defines the domain in the DNS tree whose RRSIG signatures are to be trusted. This might be the root domain (".") or a child domain ("norway.sans.org."). The trust anchor key is distributed outside of DNS and prior to any DNSSEC queries (or else there would be an infinite chicken-and-egg regress). On Windows Server 2012 and later, trust anchor keys can be distributed to all DNS servers throughout the forest via AD replication, or, on stand-alone DNS servers, the keys are in the TrustAnchors.dns file which must be copied to the other stand-alone DNS servers by hand or script. Trust anchors are not distributed to non-DNS servers, such as workstations and tablets, because clients typically do not perform DNSSEC signature validation themselves.

In the DNS manager tool, trust anchors are visible in the Trust Points container if you have checked the box to "Enable DNSSEC validation for remote responses" in the properties of the DNS server (on the Advanced tab) and do a refresh.



Desired Record ← RRSIG ← DNSKEY ← Trust Anchor Key

[Where Trust Anchor Key = A special DNSKEY replicated through AD.]

There are "the" trust anchor keys for the root of the entire DNS hierarchy, but an organization might create its own local trust anchor keys for its own portion of the DNS hierarchy alone. If an organization just uses DNSSEC for its own DNS records without any relationships to other parts of the DNS hierarchy, then that organization's trust anchor keys demarcate the apex of its own private "island of security" for DNSSEC.

Starting with a trust anchor public key, a client can check the RRSIG signature for other records in a particular DNS domain, including the RRSIG signature of the DNSKEY record for that domain, which just happens to also contain a copy of that very same trust anchor public key. But what about child domains/zones, how are they validated? A Delegation Signer (DS) record in a parent domain ("sans.org") contains the hash of a DNSKEY record for a child domain ("norway.sans.org") and that DS record is itself signed with an RRSIG record. Starting with the anchor key(s), one can validate a DS record, which validates the DNSKEY record in a child domain, which can be used to validate other DS records in the child domain for the DNSKEY records in other grandchildren domains, and so on. With signed DS records, it is possible to "walk" the validation path from an anchor key down to a particular RRSIG signature for a desired record, or to start with a desired record and "walk" the validation path up through its RRSIG record, DNSKEY record, DS record, etc. until arriving up at the top with the anchor key(s).

Record ← RRSIG ← *[DS+DNSKEY]* ← DNSKEY ← Trust Anchor Key

[Where "*[DS+DNSKEY]*" = One or more links connecting a signed DS record in a parent DNS zone referring to the DNSKEY record in a child zone.]

Incidentally, NSEC3 records are used to verify the non-existence of other DNS records, but without allowing an enumeration of all records (unlike the older NSEC records, which did allow such enumeration, which is also called "zone walking"). To make zone walking more difficult for attackers, NSEC3 records contain salted hashes of FQDNs, not plaintext FQDNs, where the length of the hexadecimal salt, the number iterations for computing the hash, and the frequency with which the salt is changed make it difficult for the attacker to compute new rainbow tables of all likely FQDNs. The salt and iteration count values themselves are in plaintext in the NSEC3 records themselves.

NSEC3 records also permit the DNS administrator to "opt-out" from being required to prove or disprove the existence or non-existence of subdomains underneath the domain which is signed. Hence, if the "testing.local" domain is signed, but the "norway.testing.local" subdomain is not signed, NSEC3 records with the opt-out flag set indicate that no promises are made about that subdomain. Enabling the opt-out option is handy when unsigned subdomains are frequently added and removed.

Key Rollover
"Rollover" is the process of generating and distributing new DNSSEC keys. On Server 2012 and later DNS servers which are domain controllers, rollover is mostly automatic and hands-free. The "Zone Signing Key (ZSK)" is the public-private key pair which is used to create RRSIG records. The "Key Signing Key (KSK)" is the public-private key pair used to sign the ZSK. Now, whether the ZSK and the KSK are different key pairs or the same key pair, their sizes, lifetimes, and so on are all operational details which the DNSSEC specifications do not govern. Windows Server uses different ZSK and KSK key pairs to assist with rollovers, but this is not required by the RFCs. The public keys of the ZSK and KSK pairs are made available as DNSKEY records, but the private keys of the ZSK and KSK pairs are never accessible through DNS queries.

Your DNS Registrar
However, key rollover is not so simple when one's DNS zone is accessible to the Internet and you want to provide DNSSEC validation to anyone in the world. First, you'll need to contact the registrar of your parent DNS zone (which is probably the registrar for .com, .edu, .net or .mil) and find out how they handle the process of receiving, authenticating, managing and receiving payment for the hosting of your DS records. Your DS records in the registry of your parent DNS zone is how the world can chain from the signed root zone (".") down to your particular domain, validating all the necessary records and keys along the way. Well, your DNS parent won't just allow any random person to declare that they are authoritative for a child zone and present any random DNSKEY or DS record. There must be a process for authenticating you as the legitimate administrator of the child zone, a process for conveying your DNSKEY/DS information to the parent, and a process for the graceful rollover of your old keys to your new keys. Your registrar will likely support Extensible Provisioning Protocol (EPP) for DNSSEC (RFC5910), but using tools which can be scripted for scheduled tasks? For all these issues, this manual is just going to punt. We just don't have the time and everyone's parent registrar will have different procedures anyway. If you wish to link your DNSSEC implementation to the rest of the world, contact your DNS registrar and download their guidance and procedures (and hope that they have these procedures for you, or else you may need to transfer your zones to another registrar). To find a DNSSEC-friendly registrar, do an Internet search on "registrars that support dnssec", which will lead, for example, to registrars like www.GoDaddy.com and www.Dyn.com.

Troubleshooting, News, Tutorials and RFCs
Some nice web sites for testing and understanding DNSSEC are:

- http://dnsviz.net (test DNS zones)
- http://dnssec-debugger.verisignlabs.com (test DNS zones)
- http://dnssectest.sidnlabs.nl (client browser test)
- http://test.dnssec-or-not.com (client browser test)
- https://www.dnssec-validator.cz (browser extension)
- http://www.dnssec.net (references)
- http://www.dnssec-deployment.org (news)

The following DNSSEC RFCs should be read in this order, especially the first one:

- RFC 4033: DNS Security Introduction and Requirements
- RFC 4034: Resource Records for the DNS Security Extensions
- RFC 4035: Protocol Modifications for the DNS Security Extensions
- RFC 2535: Domain Name System Security Extensions
- RFC 4631: DNSSEC Operational Practices

DNS Client Behavior (Be Careful What You Ask For...)

When a DNS client queries for a record and DNSSEC is not required for that record by the client, then there is no change in behavior. It doesn't matter whether there are DNSSEC signatures or not. (This leaves the client vulnerable to MITM and spoofing attacks for those records, though, which is the whole point of using DNSSEC.)

When a DNS client queries for a record and DNSSEC is required for that record by the client, if the DNS server's response indicates successful validation in its response, then there is no change in behavior, the user sees nothing different.

But if for any reason the server does not set the flag in its response indicating successful validation, then the client will behave as though DNS name resolution had failed entirely, and the user will see an error message in whatever application the user was running which needed the name to be resolved, like their browser or e-mail client. But this error is what we want, namely, for the user to be denied access to servers when DNSSEC fails to authenticate that server's DNS records. Hence, be careful for which FQDNs, domains and IP subnets you *require* DNSSEC validation, or else you might inflict a self-imposed DoS attack.

> **Note:** When directly querying an authoritative DNS server for a particular zone, such as your own DNS server for the testing.local zone, then the response is always considered valid. So even if you both unsign all records in testing.local and require DNSSEC validation for them, you will not get an error message during resolution because you happen to be querying the authoritative DNS server for that zone. This is expected behavior, but it can make it more difficult to do testing and demos with just one VM that has no Internet access.

To test name resolution from a client with various DNSSEC options using PowerShell:

```
Resolve-DnsName -Name testing.local -Server 127.0.0.1 -DnssecOK

# DNSSEC Parameters:
# -DnssecOK = Client is DNSSEC-aware, OK to send RRSIG records.
# -DnssecCD = DNSSEC checking disabled on the client.
# -DnsOnly = Client only uses DNS to resolve, not LLMNR/NetBIOS.
```

Keep in mind, though, that if an application implements its own DNSSEC rules, such as Mozilla Firefox with various DNSSEC add-ons, then the behavior of that application can be different than other applications which rely solely on the operating system.

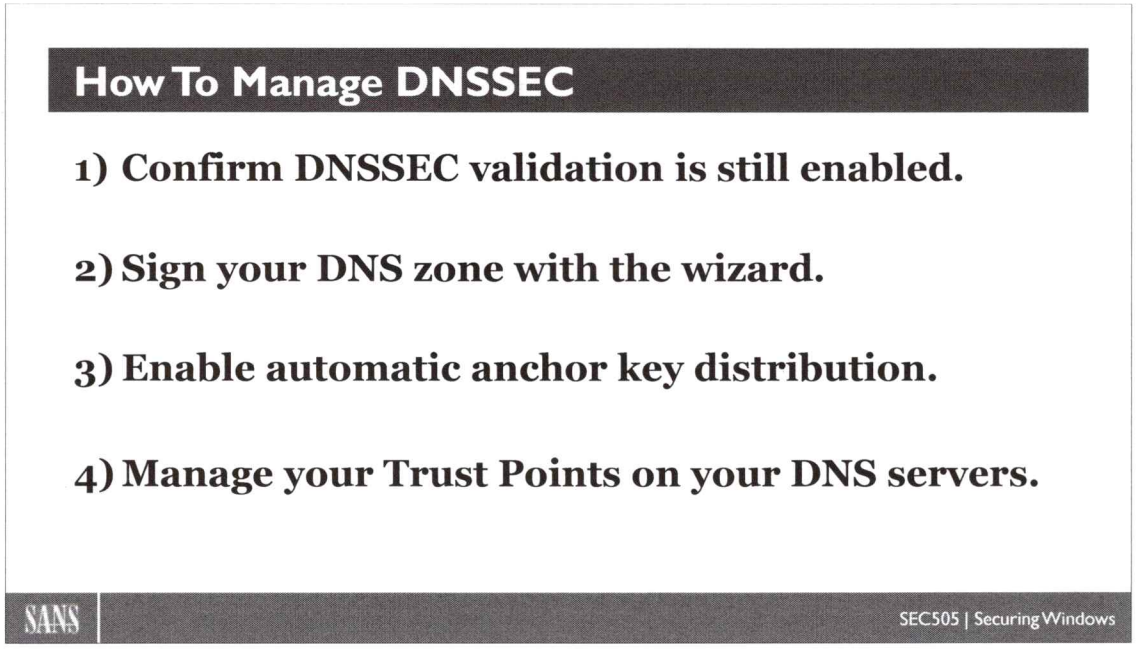
Do Windows Clients Check DNS Signatures Too?

No. Windows 2000 through Windows XP, and Server 2000 through Server 2008-R1, do not use or support DNSSEC in any way.

And as DNS clients, Windows 7, Server 2008-R2 and later do not check DNSSEC signatures themselves. However, these clients can indicate in their queries to their DNS servers that they are "DNSSEC aware" (the DO flag in the query), which means that they can look for another flag set by the DNS server (the AD flag in the response) when it resolves and validates a name successfully using DNSSEC. In other words, the DNS server attaches a yellow Post-It Note to its response back to the client which says, "Don't worry, I checked the signature for you." Needless to say, this little flag can be spoofed by attackers too, hence, IPSec would still be necessary to secure the queries and responses between internal clients and internal DNS servers.

The benefit of using non-validating but DNSSEC-aware clients is 1) reduction of burden on the clients and 2) administrators do not have to distribute and update trust anchors to the clients. Trust anchor keys only need to be distributed and updated on the DNS servers when clients do not perform DNSSEC signature validation themselves. But if IPSec is configured on the clients for DNS traffic, then this imposes a performance and management burden itself. Time will tell whether client-side DNSSEC validation will become an available option.

Some applications, however, support client-side DNSSEC verification of signatures even if the operating system does not; for example, there are DNSSEC add-ons for Mozilla Firefox, and DNSSEC may become a built-in feature for both Firefox and Google Chrome. Internet Explorer, on the other hand, will rely on the operating system for its DNSSEC integration, assuming that client-side DNSSEC signature checking is ever added at all.



How To Manage DNSSEC

Managing DNSSEC is relatively easy on Server 2012 and later, especially when the DNS service is installed on a domain controller. DNSSEC works on stand-alone DNS servers too, but the management is more labor-intensive. Installing the DNS service on domain controllers is recommended for the sake of secure dynamic updates anyway, so this section only discusses DNSSEC when DNS is installed on domain controllers.

Here are the tasks that need to be performed to enable and configure DNSSEC.

Confirm That DNSSEC Validation Is Still Enabled (The Default)

DNSSEC validation is enabled by default on Server 2012 and later DNS servers.

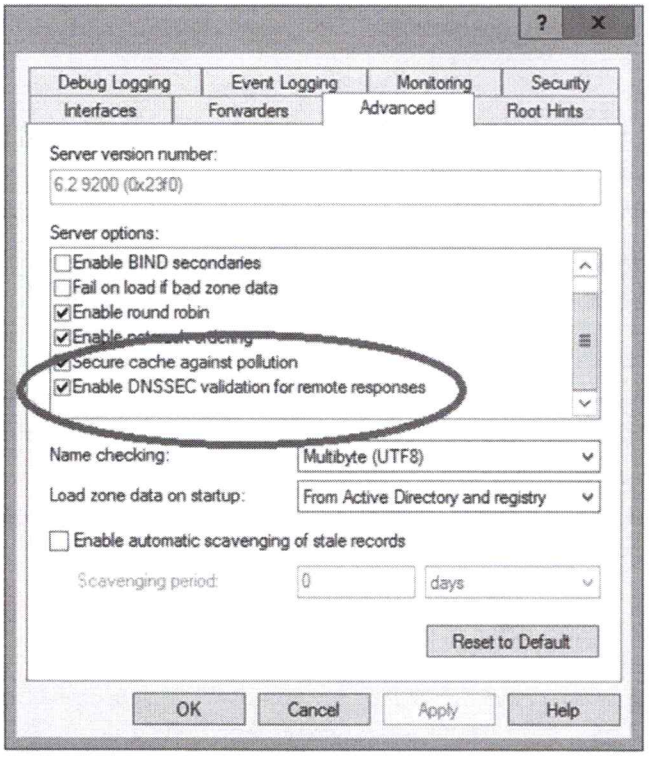
To query the current status of DNSSEC validation on a local or remote DNS server:

```
Get-DnsServerSetting -Computer LocalHost | Select EnableDnsSec
```

To enable or disable DNSSEC response validation on your DNS server:

```
DnsCmd.exe /Config /enablednssec 1          #Enable

DnsCmd.exe /Config /enablednssec 0          #Disable
```

On Server 2012, this setting is exposed in the graphical DNS snap-in. In the DNS manager tool, right-click your DNS server, go to Properties, Advanced tab, then confirm that the "Enable DNSSEC validation for remote responses" checkbox is checked. (This checkbox does not exist on Server 2016 or later.)



This checkbox is found on Server 2012, but not on Server 2016 or later.

It's important to know how to quickly turn off DNSSEC validation in an emergency if DNS name resolution suddenly stops working correctly.

Sign Your DNS Zone

Enabling DNSSEC for your own DNS records begins with the creation of trust anchor keys to sign the zone. Trust anchor keys are the initial keys used with DNSSEC to sign other DNS records.

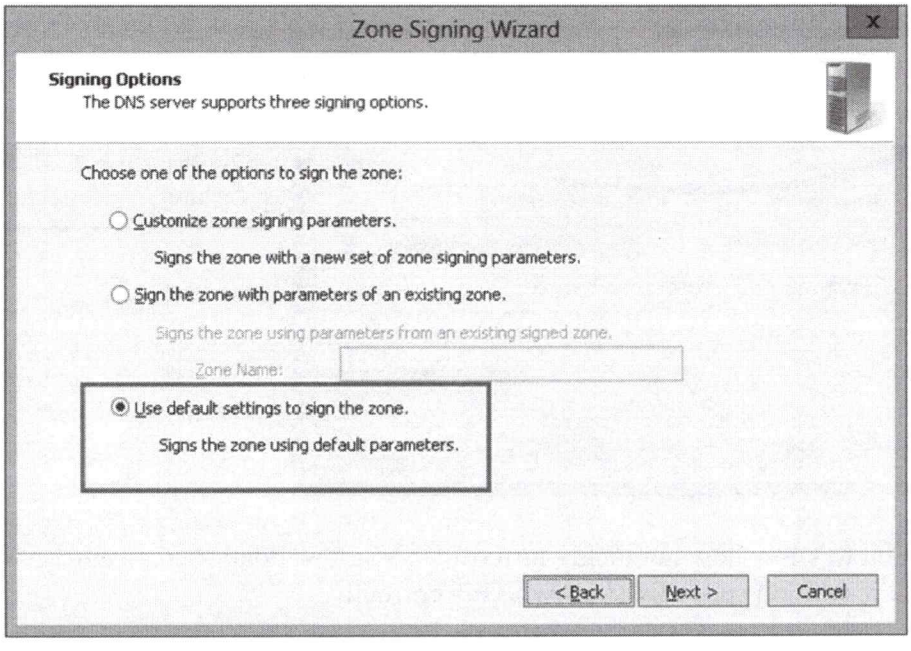
You only have to sign the records in your DNS zone once. New or modified records will be automatically signed as needed. This is compatible with file-based and Active Directory-integrated zones, even when dynamic updates is enabled. Strictly speaking, a DNS record is not itself signed, which would imply that that record is modified in some way; instead, new DNS records are added (such as RRSIG records) which contain the signature data for your other previously-existing records. Hence, "signing the zone" means simply adding more DNS records, not modifying your existing ones.

Signing your DNS zone is done by a graphical wizard. While it is possible to customize the zone signing parameters when you initially sign the zone, this will rarely be necessary. Besides, you can always edit these parameters afterwards anyway without being forced to resign everything, such as changing trust anchor distribution settings. The default settings will almost always be sufficient. With good intentions you might increase the signing key sizes or choose different hashing algorithms, but beware of the

performance penalties and cross-platform incompatibilities this might cause, so only change the defaults if you know what you are doing.

Try It Now!

To sign a DNS zone, right-click on the desired domain > DNSSEC > Sign the Zone > Next > Use default settings to sign the zone > then click Next repeatedly as necessary many times > Finish.



After signing your DNS zone, refresh your list of records in the DNS snap-in and notice that you now have many new records of types RRSIG, DNSKEY and NSEC3. There is also a padlock icon on top of the yellow zone container for the DNS domain.

A zone can be "unsigned" as well, which means all of its DNSSEC-related records will be deleted, including the trust anchor keys, but not your other regular DNS records. Just right-click the domain > DNSSEC > Unsign the Zone. (But don't do that now.)

To display the current DNSSEC settings for a DNS zone:

```
Get-DnsServerDnsSecZoneSetting -ZoneName <Your-DNS-Domain-Name>
```

Enable Automatic Key Distribution

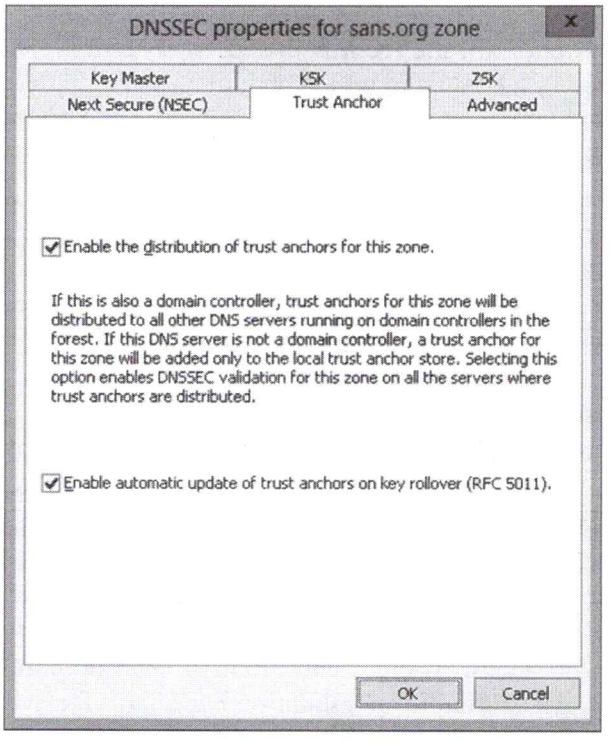
Trust anchor keys are used to sign DNS records and other DNSSEC-related keys. Every DNS server inside the LAN must have copies of these keys in order to perform DNSSEC validation.

Trust anchor key distribution can be made automatic on domain controllers. Hands-free DNSSEC key distribution is one of the major benefits of using Active Directory-

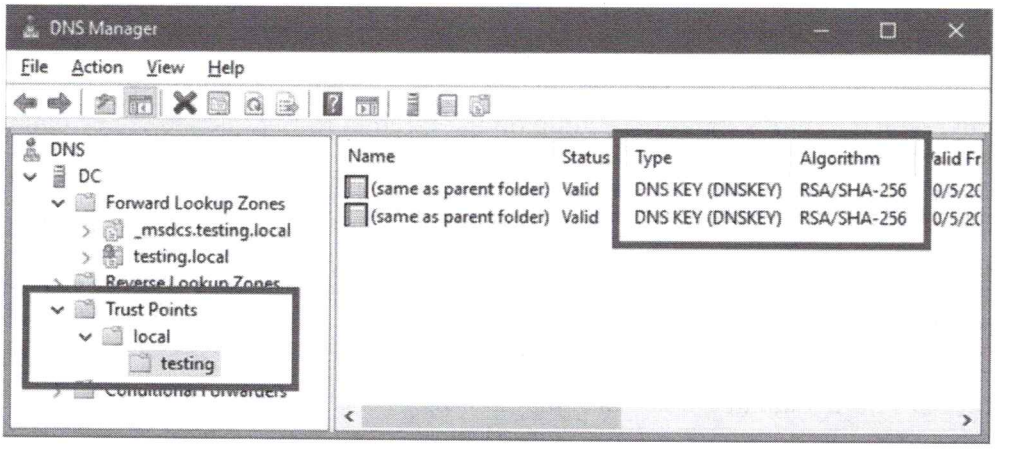
integrated DNS. If you have DNS server which are not AD-integrated, the keys must be copied by hand or scheduled task.

Try It Now!

To enable automatic trust anchor distribution to all domain controllers in the forest, open the DNS Manager tool > right-click your DNS domain > DNSSEC > Properties > Trust Anchor tab > check "Enable the distribution of trust anchors for this zone" > OK > Yes.



Once enabled, click the refresh button in the toolbar of the DNS snap-in and you'll see the Trust Points container populated with your DNSSEC information. There are two DNSKEY records to be seen --the active and the standby keys-- and these will be replicated to your other domain controllers via Active Directory.



To confirm the availability of trust anchor records in PowerShell:

```
Get-DnsServerTrustAnchor -Name YourDomainName | fl *
```

Manage Your Trust Points (Or Else!)

There is something very important you should know about this innocent-looking Trust Points yellow container in the DNS snap-in, and Microsoft does not do a good job about warning us about the risks.

Enabling DNSSEC validation for the records in a DNS zone requires 1) that validation be enabled for the DNS server as a whole, which we did earlier in the manual, and 2) that DNSKEY records are added for at least one zone under the Trust Points container, i.e., the Trust Points container must be visible and not empty. Meeting these two criteria for a particular zone does not just enable DNSSEC validation for records in that zone, as though it were merely preferred, but makes successful validation of those records *mandatory*. Mandatory validation means that DNS name resolution will fail for any domain listed under Trust Points if DNSSEC validation does not succeed for some reason. Be careful what you add to the Trust Points container!

Root Zone Self-Imposed DoS Attack

For example, imagine that you add the DNSKEY records for the root zone in DNS. The root DNS zone, which is represented by a single period ("."), is the top-most domain under which all other DNS subdomains are categorized, such as "sans.org." (notice the final period in "sans.org.", it's not a typo). If you were to add the root zone anchor keys to the Trust Points container, then every DNS record from every possible DNS domain must be signed with DNSSEC, and your DNS server will deliberately fail to resolve any name which cannot be validated. There are millions of DNS domains which are not yet signed with DNSSEC keys, so requiring validation by default for everything will result in a massive self-imposed Denial of Service attack. So, be careful what you add!

Adding Trust Points

How do you add to the Trust Points container? In other words, how do you add DNSKEY records for a particular DNS zone? This can be done in three ways: 1) automatic anchor key distribution through Active Directory, 2) using the graphical DNS snap-in tool, or 3) scripting in PowerShell.

Here is an example of using PowerShell to add the DNS root zone anchor keys on Server 2012 only, which requires first researching on the Internet to find the latest key data:

```
# C:\SANS\Day6-Servers\DNS\Add-RootZoneTrustAnchor.ps1

Add-DnsServerTrustAnchor -Name . -CryptoAlgo RsaSha256 -Digest
49AAC11D7B6F6446702E54A1607371607A1A41855200FD2CE1CDDE32F24E8FB5
-DigestType Sha256 -KeyTag 19036
```

On Server 2012 R2 and later, Microsoft made adding root zone keys easier:

```
Add-DnsServerTrustAnchor -Root          #Server 2012 R2 and later
```

Both of the above commands require Internet access so that your DNS server can talk to other DNS servers around the world as necessary to get the DNSKEY records needed.

In the DNS snap-in, you can also right-click the Trust Points container and select Add or Import, but you'll have to research what key data you need first. For example, let's say you wanted to require DNSSEC validation for the sandia.gov domain, which does publish DNSKEY records to the Internet. The www.dnsviz.net web site is wonderful for understanding and troubleshooting DNSSEC. Using the dnsviz.net web site, you run a query for "sandia.gov" and you click the link to go to this page (or just go right there):

http://dn-sviz.net/d/**sandia.gov**/responses/

On this page, find the Delegation Signer (DS) records for the sandia.gov domain. You will find information similar to the following:

Responses for sandia.gov/DS			
Key Tag	**Algorithm**	**Digest Type**	**Digest**
20739	7	1	3ca461ff5496bc72a772056489d944621eda774e
20739	7	2	0c5f4cdff8824665acd1d8a132951b193c59fa7fe6cff824...

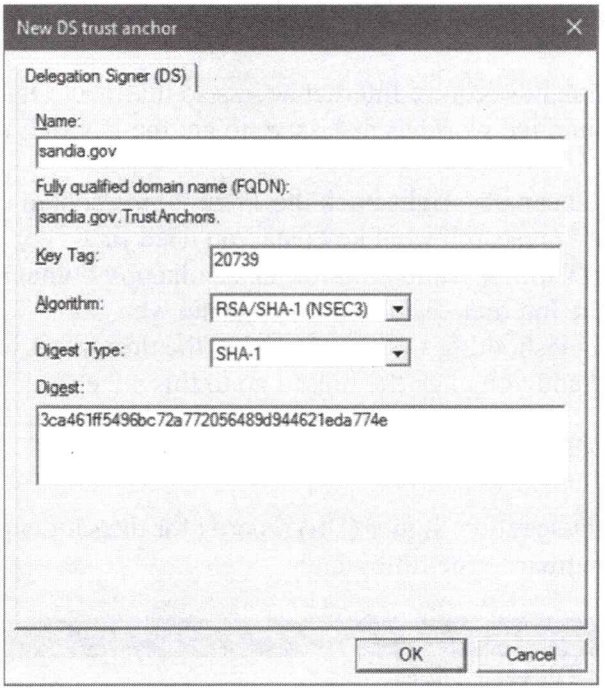
From RFC6014 and the good people at IANA we know that these are the algorithm combinations used on Windows Server:

http://www.iana.org/assignments/dns-sec-alg-numbers	
Algorithm Number	**Cipher Suite**
13	ECDSAP256/SHA-256
14	ECDSAP384/SHA-384
5	RSA/SHA-1
7	RSA/SHA-1 (NSEC3)
8	RSA/SHA-256
10	RSA/SHA-512

And these are the digest type numbers:

http://www.iana.org/assignments/dns-sec-alg-numbers	
Digest Type Number	**Digest Type Name**
1	SHA-1
2	SHA-256
4	SHA-384

Using the above information, you may right-click the Trust Points container in the DNS snap-in, select Add, choose DS, and enter the data shown in the following screenshot:



Under the Trust Points container, the entry at first will say "DS Pending", then it will soon show the DNSKEY records that were downloaded for sandia.gov instead. Your DNS server will now require successful DNSSEC validation for all records within that domain. You only have to add one DS entry, by the way, not all of the DS information available for a domain, your DNS server will figure the rest out for itself.

The same DS record information for sandia.gov can be added through PowerShell:

```
Add-DnsServerTrustAnchor -Name sandia.gov -CryptoAlgorithm
RsaSha1NSec3 -DigestType Sha1 -Digest
3ca461ff5496bc72a772056489d944621eda774e -KeyTag 20739


Add-DnsServerTrustAnchor -Name sandia.gov -CryptoAlgorithm
RsaSha1NSec3 -DigestType Sha256 -Digest
0c5f4cdff8824665acd1d8a132951b193c59fa7fe6cf7b1f82484c25b410cdc6
-KeyTag 20739


Add-DnsServerTrustAnchor -Name sandia.gov -CryptoAlgorithm
RsaSha1NSec3 -DigestType Sha1 -Digest
e33b4526cecf2a1b7c733d645b30cd5c912d9538 -KeyTag 36033


Add-DnsServerTrustAnchor -Name sandia.gov -CryptoAlgorithm
RsaSha1NSec3 -DigestType Sha256 -Digest
4677626abe69fd1d8dd8e5de10533d2b264a91a7bf1ab3a6bffc4c408a3752ff
-KeyTag 36033
```

When Unsigning Zones
When you unsign your DNS zone, don't forget to delete the trust anchor keys from the Trust Points container in the DNS snap-in. Just right-click on the DNSKEY records under Trust Points and select Delete. When you do a refresh, you'll see that the zone disappears entirely from under Trust Points.

Manual Anchor Key Distribution (When Not AD-Integrated)
On DNS servers which are not domain controllers the trust anchor key files must be copied manually from a folder on the DNS server where they were created to your other DNS servers. This must be done by hand or scheduled script, and it must be repeated whenever new keys are generated. This is a major pain.

The trust anchor files are stored on the hard drive of the DNS server in the C:\Windows\System32\dns folder, and this folder can be safely shared read-only to the Everyone group, perhaps shared as "dnskey." Once this folder is shared, then, on a DNS server which is not a domain controller, the trust anchor files can be imported from the share. If the DNS name of one's domain is "testing.local", then the DNSKEY information is stored in a file named "keyset-testing.local".

> ***Try It Now!***
>
> To import trust anchor files on a DNS server which is not a domain controller, open the DNS Manager tool > right-click Trust Points > Import > DNSKEY > navigate to the correct shared folder on the other DNS server, e.g., \\servername\dnskey > select the correct "keyset-*" file > OK.

Automatic distribution of trust anchor keys to non-authoritative DNS servers only occurs when there is a key rollover. If you re-sign the domain with a new key, you'll have to distribute the new trust anchor manually again, which should be done immediately. If a DNS server has the wrong trust anchor records, DNS queries that require validation will return errors. In some cases, it will be easier to simply delete all the DNSKEY records previously imported from the other DNS server, clear the local server and client name resolution caches, and import that server's DNSKEY records again.

Transfer The Key Master Role

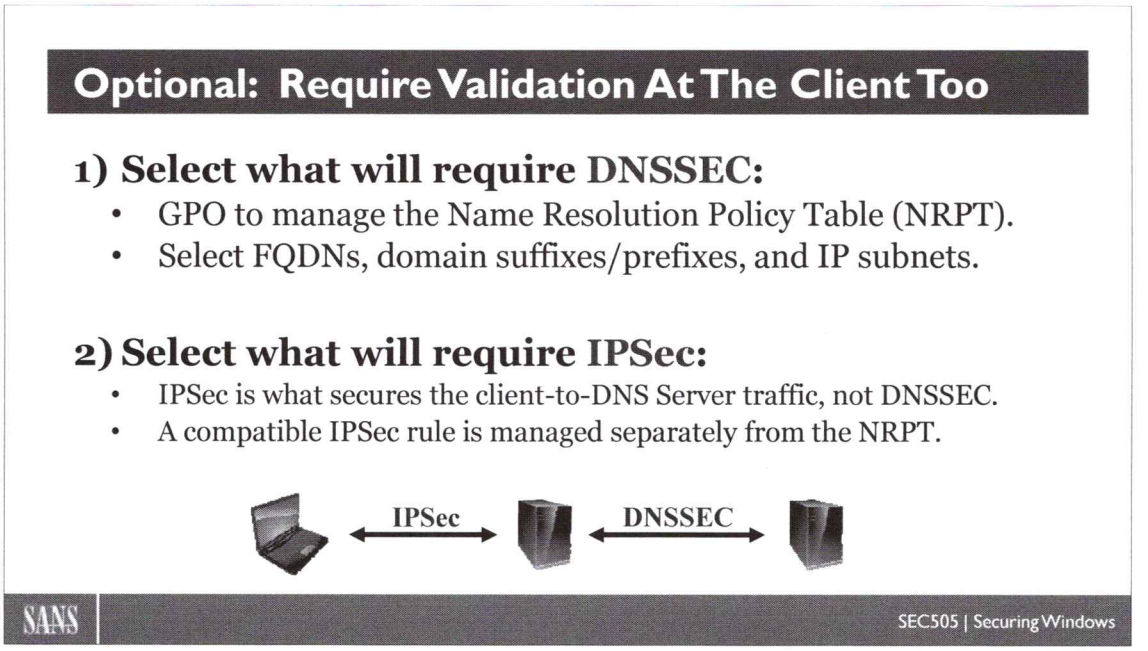
The Key Master is that DNS server which currently manages the encryption keys for the DNSSEC zone. There can be only one Key Master. (The DNS Key Master is not an example of an Active Directory operations master, such as the Schema Master.) Any primary DNS server can become the Key Master if you wish to transfer the role. The Key Master does not have to be a domain controller or host any Active Directory-integrated DNS zones. Whatever DNS server is chosen for this role, that server should be well-guarded inside the LAN and continuously monitored for problems.

> ***Try It Now!***
>
> To transfer the Key Master role from one primary DNS server to another, open the DNS manager tool > right-click the domain > DNSSEC > Properties > Key Master tab > pull down the menu to select the new server > OK.

Multi-Forest Environments and Conditional Forwarding

In a multi-forest environment, the DNS servers in your AD forests will almost certainly have conditional forwarders configured in DNS so that every DNS server in every forest can forward queries to the other DNS servers correctly. Conditional forwarding has been around for many years, so this is nothing new, but it can get complicated when DNSSEC is added because many (or all) of these DNS domains are isolated from the outside world, thus creating "DNSSEC islands" which cannot be validated by starting at the DNS root zone and working down to all of one's local domains.

There is no cross-forest anchor key automatic distribution. This is true even with two-way cross-forest trusts. Hence, after conditional forwarders have been configured in DNS, the DNSKEY information from each forest must be imported into the DNS servers of the other forests. This can done using the same steps described above in the *Try It Now* exercise. If this import is not done and DNSSEC validation of remote responses has been enabled in the properties of one's DNS servers, then DNS name resolution will fail.



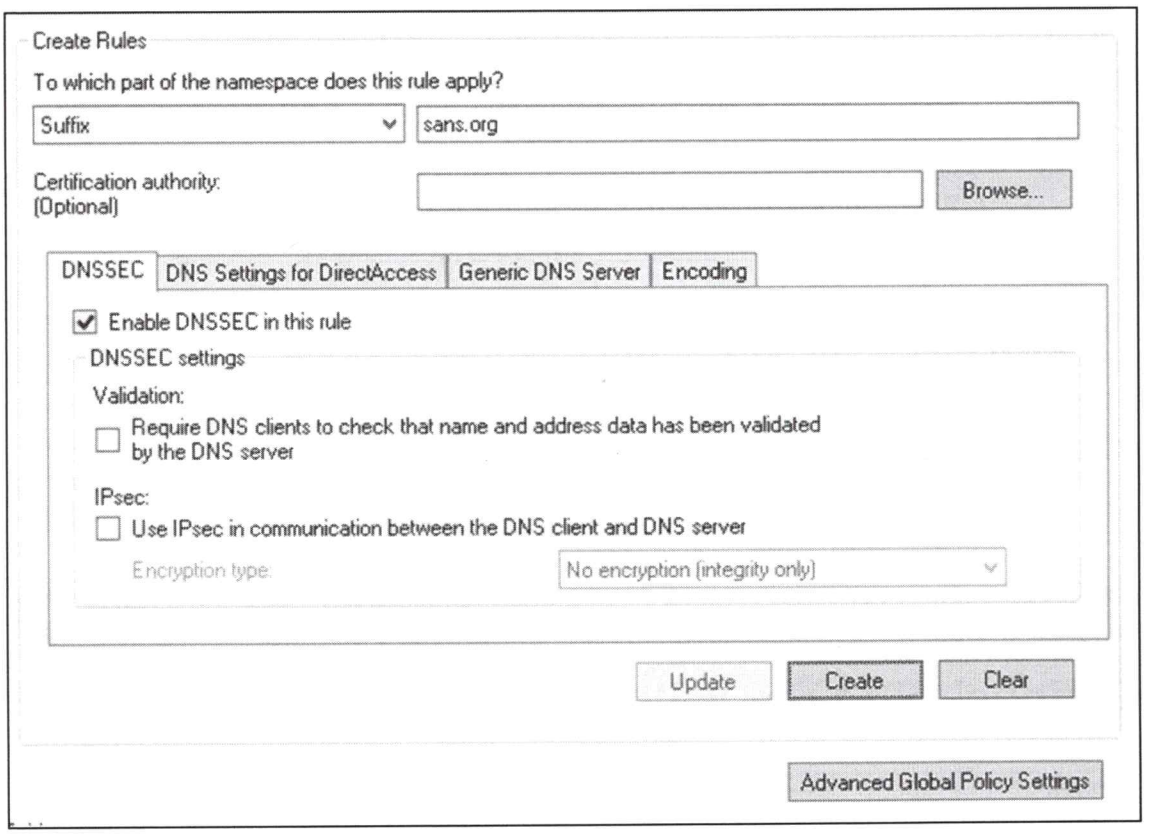
Optional: Require Validation At The Client Too

When a client requests DNS name resolution from a server, how does the client know when to require DNSSEC from the DNS server? When is DNSSEC optional and when is it mandatory? The client's DNSSEC settings are controlled by its Name Resolution Policy Table (NRPT). Keep in mind, though, that only Windows 7, Server 2008-R2 and later operating systems support DNSSEC or have an NRPT.

The NRPT determines which DNS domains, if any, will require DNSSEC validation. It is also used for DirectAccess, but we'll ignore DirectAccess here. The NRPT is managed through Group Policy and its options in a GPO are located under Computer Configuration > Policies > Windows Settings > Name Resolution Policy.

The NRPT on a computer will contain zero or more rules. Each rule enables or disables the requirement for DNSSEC validation for a set of names. The set of names in a rule could be:

- Just a single FQDN (vpn.sans.org)
- A prefix with an implicit wildcard after it (vpn.sans.*)
- A domain suffix with an implicit wildcard before it (*.sans.org)
- An IPv4/IPv6 subnet (10.4.0.0/16)
- An "Any" wildcard to match every possible name (*)



Some of these rules are mainly for doing reverse DNS lookups with pointer records (prefix and subnet) and the "Any" wildcard is reserved for roaming DirectAccess clients who tunnel 100% of their traffic through the office LAN. When configuring DNSSEC options, you'll mainly use individual FQDNs (such as "vpn.sans.org") and domain suffixes (such as "sans.org", which is treated as "*.sans.org", but you don't type the asterisk when configuring the rule, it's implied).

If the name to be resolved matches multiple NRPT rules, the rules are ranked from most to least preferred. The numbering in the list of sets in the dialog box above is the order of ranking, hence, a FQDN rule is preferred over a prefix rule, and the "Any" at the bottom is the least preferred. If two suffix rules match the name to be resolved, whichever suffix is longer in the rule is the preferred one, hence, when resolving "vpn.norway.sans.org", this would match rules for both "sans.org" and "norway.sans.org", but the latter is longer and so it would be preferred.

Try It Now!

To create an NRPT rule to require DNSSEC validation for all names which match the "*.norway.sans.org" suffix, edit the desired GPO > Computer Configuration > Policies > Windows Settings > Name Resolution Policy > select "Suffix" for the part of the namespace to which the rule applies > enter "norway.sans.org" beside the Suffix (notice, there is no "*") > check the box "Enable DNSSEC in this rule" > check "Require DNS clients to check that the name and address data has been

validated by the DNS server" > Create button (which moves the rule in the NRPT table at the bottom) > Apply.

Note: If you are testing this, remember to run "gpupdate.exe /force" afterwards.

Important: When multiple GPOs apply to a computer, all the NRPT rules from all the GPOs are merged together. However, if two rules have the exact same namespace target, e.g., FQDN or domain suffix, then neither rule will be applied.

Important: Any NRPT rules you create in the local GPO will be ignored if even a single NRPT rule is applied through a GPO from Active Directory.

PowerShell NRPT

The NRPT can also be managed through PowerShell cmdlets:

```
get-help *nrpt*
```

To view the current DNSSEC requirements, if any, in your local NRPT:

```
# Requires Windows 8, Server 2012, or later:
Get-DnsClientNrptRule

# For Windows Vista, Windows 7, and Server 2008:
netsh.exe namespace show policy
```

To view the NRPT rules from any source currently in effect:

```
Get-DnsClientNrptPolicy -Effective
```

Require DNSSEC validation for "www.sandia.gov" FQDN in the local GPO:

```
Add-DnsClientNrptRule -DnsSecEnable -DnsSecValidationRequired
  -Namespace "www.sandia.gov"
```

Require DNSSEC validation for the ".sandia.gov" domain suffix in the local GPO:

```
Add-DnsClientNrptRule -DnsSecEnable -DnsSecValidationRequired
  -Namespace ".sandia.gov"
```

Notice the beginning period (".") in the prior command for the ".sandia.gov" domain. Without this beginning period, the namespace given will be interpreted as a FQDN.

Remove all NRPT rules from the local GPO:

```
Get-DnsClientNrptRule | Remove-DnsClientNrptRule -Force
```

If you want to see changes made to the local GPO by the commands above, close the MMC.EXE console showing the local GPO and open it back up again.

> **Note:** Using NRPT rules, you can query different DNS servers for different FQDNs or domains (see the Generic DNS Server tab in this GPO dialog box).

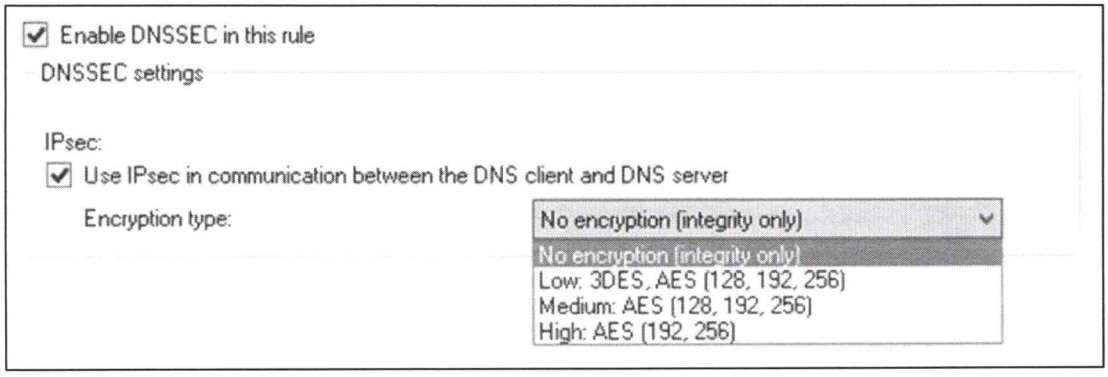
Create Exceptions for FQDNs and Subdomains

Because of the preference ordering of overlapping rules, you can define an exception for a single FQDN or a particular subdomain as necessary. When defining exceptions where DNSSEC should not be required, define the rule, enable DNSSEC for it, but do not check the box named "Require DNS clients to check that the name and address data has been validated by the DNS server." Make sure the exception rule has a ranking number which will make it more preferred than any other matching rule which would require DNSSEC.

Why Are There IPSec Options When We're Using DNSSEC Already?

Recall that DNS clients do not actually check any DNSSEC signatures, they only check a flag in the DNS server's response which tells the client that the DNS server did the signature checking for them. But this flag can be modified in transit and attackers can spoof DNS responses with the flag set as the attacker wishes. Microsoft does not emphasize this much, but keep in mind that only Windows DNS servers can actually check DNSSEC signatures, not Windows clients to DNS, i.e., the stub resolvers.

So the only way a client can *really* know it is communicating with the intended DNS server, and the only way a client can really know that no DNS responses have been tampered with, is to use IPSec. IPSec can sign and/or encrypt all DNS queries and responses with your own DNS servers (but not with other DNS servers on the Internet). There is a performance penalty when using IPSec with DNS, though, as we'll discuss in the IPSec portion of this course. You'll have to decide what is more important to you after you've done your testing (and perhaps upgraded the NIC on your test DNS server to support IPSec Offload).



When configuring an NRPT rule, you can check the box to use IPSec for DNS and set the security level. For performance reasons, it's best to choose the "No encryption (integrity only)" option. What this means and how to manage IPSec is discussed elsewhere in this

course at length, so we won't cover it here. (Incidentally, the certification authority option in the NRPT is for certificate-based IPSec, not for DNSSEC.)

Testing DNSSEC In The Lab

Testing DNSSEC in the lab is a bit of challenge when you only have one DNS server. When you query a DNS server which is authoritative for the record you are testing, the server's response does not include the trust-me-I-checked-signature-for-you flag (the AD flag) even if it has a DNSSEC signature for that record. This might seem crazy, but the DNS server has to trust itself and it is authoritative for the DNS zone being queried.

And even when a DNSSEC-aware client has a NRPT which mandates the use of DNSSEC validation for a particular name, if that name is resolved by querying an *authoritative* DNS server for that name, not just some random recursive DNS server at an ISP, then the DNSSEC requirement is waived and the resolution will succeed even without the AD flag being set in the server's response. Again, this might seem crazy, but a client has to trust authoritative DNS servers for a domain or else there's no point to DNSSEC at all! How does the client know that the queried DNS server is authoritative? The server said so, by setting a flag for this purpose (AA) in its response...(ahem).

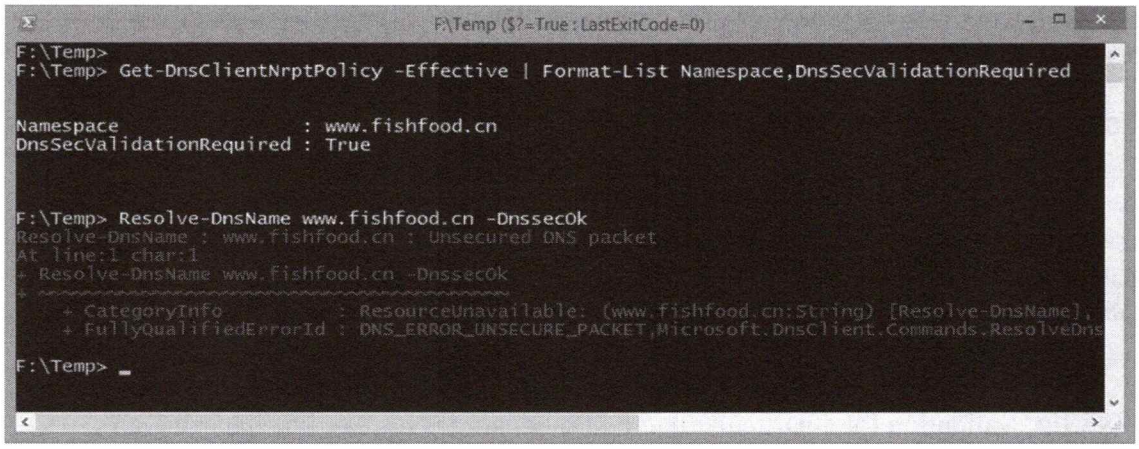
But, at least you can query for the RRSIG and DNSKEY records directly in the lab. If your host computer has Internet access, you can perform a query through the public DNS servers for Google (8.8.8.8) or OpenDNS (208.67.222.222):

```
Resolve-DnsName -Server 8.8.8.8 -Name com. -Type SOA -DnsSecOK

Resolve-DnsName -Server 8.8.8.8 -Name com. -Type DNSKEY -DnsSecOK

Resolve-DnsName -Server 8.8.8.8 -Name iana.org -Type RRSIG
   -DnsSecOK
```

You can also get the Windows version of DIG.EXE from the Windows version of BIND, which provides the "+dnssec" switch when making queries and shows many more details of both the query and the response (https://www.isc.org/downloads/bind/).

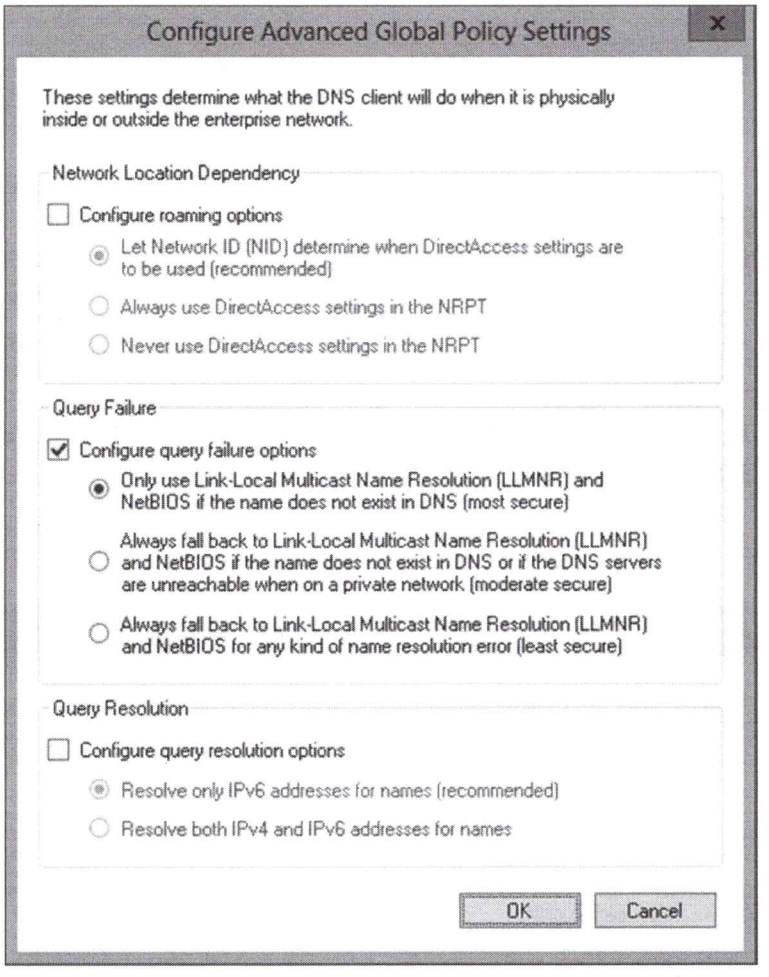
```
cmd.exe /c 'dig.exe @8.8.8.8 www.iana.org +dnssec +multiline'
```

If your lab VM also has Internet access, then configure the NRPT in the Default Domain GPO to require DNSSEC validation for a few FQDNs, then try to resolve those names; for example, try to resolve "www.iana.org" and "www.bing.com" and note what it looks like when DNSSEC validation fails. When DNSSEC validation is successful, then there really is nothing different to see (unless you use DIG.EXE and look for the AD flag in the response). But when DNSSEC validation fails, the name will appear to be unresolvable with tools like PING.EXE, and the Resolve-DnsName cmdlet will report an "Unsecured DNS packet" error message as well.



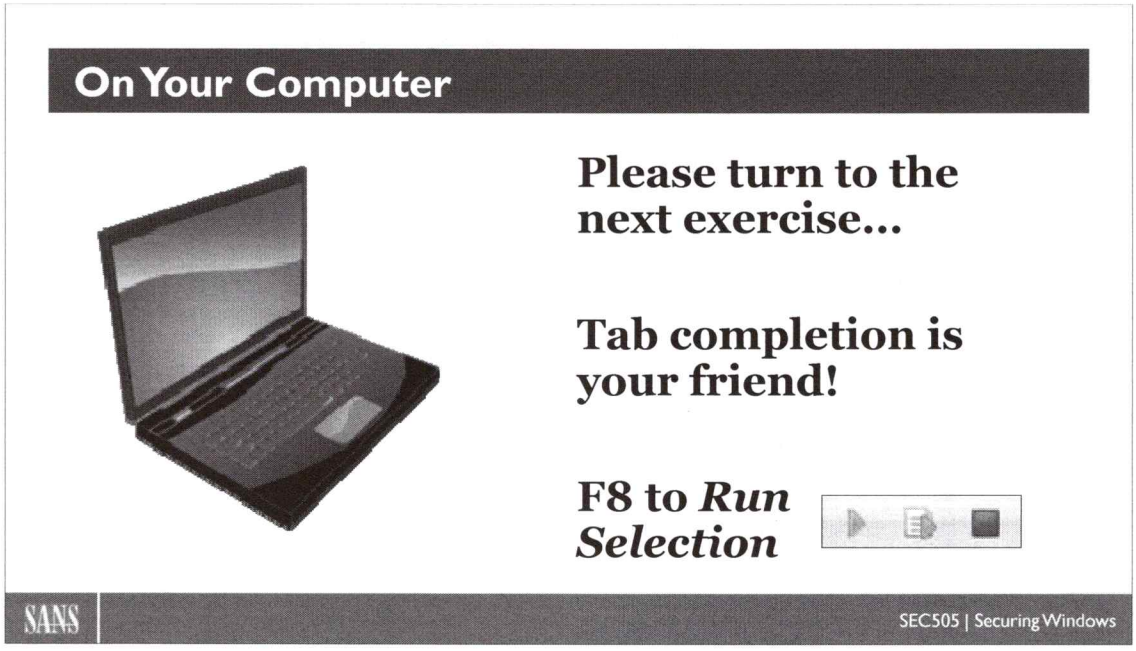
Advanced Global Policy Settings

When you click the button named "Advanced Global Policy Settings", there is a system-wide option on how to handle DNS query failures. For non-portable computers, it's best for security to choose the option named "Only use Link-Local Multicast Name Resolution (LLMNR) and NetBIOS if the name does not exist in DNS." These workstations and servers are always inside the LAN, presumably with access to multiple, reliable, internal DNS servers; hence, there shouldn't be a need for these internal-only clients to attempt any other name resolution method.



However, on portable computers which may sometimes be at the user's home or at another office, it is acceptable to choose the second option named "Always fall back to Link-Local Multicast Name Resolution (LLMNR) and NetBIOS if the name does not exist in DNS or if the DNS servers are unreachable when on a private network" (underlining added). The Windows Firewall categorizes networks as Public, Private or Domain. So, this option does not use LLMNR on Public interfaces, when it's most dangerous, or on Domain interfaces, when internal DNS servers should be reachable. But when users are at home using a Private interface, it is convenient to use LLMNR.

The third option named "Always fall back to Link-Local Multicast Name Resolution (LLMNR) and NetBIOS for any kind of name resolution error" is the least secure because attackers might deliberately corrupt or spoof DNS responses in order to trick a client into attempting LLMNR or NetBIOS name resolution, which are less secure methods than DNSSEC, and then spoof or modify LLMNR or NetBIOS responses to the client. This third option applies not just to Private-categorized interfaces, but to Public interfaces too.



On Your Computer

This lab has multiple parts. Don't forget to use tab completion!

Enable DNSSEC On Server

If you have already signed your DNS zone, then sign it again with new keys:

```
Invoke-DnsServerZoneSign -ZoneName testing.local
   -DoResign -Force
```

If you have not already signed your DNS zone, then sign it with the default settings:

```
Invoke-DnsServerZoneSign -ZoneName testing.local
   -SignWithDefault -Force
```

Enable automatic anchor key distribution through Active Directory replication:

```
Set-DnsServerDnsSecZoneSetting -ZoneName testing.local
  -DistributeTrustAnchor DnsKey
```

Refresh the view in the DNS snap-in and browse your current DNSSEC records and Trust Points for the testing.local domain. You should see your new DNSKEY, RRSIG and NSEC3 records. Double-click any of these new records to see the details, such as the cipher and hashing algorithms.

Add a DNS host record for the sake of testing:

```
Add-DnsServerResourceRecord -A -ZoneName testing.local
   -Name testhost -IPv4Address 10.1.1.5
```

Confirm that the new DNS record can be successfully resolved (replaces nslookup.exe):

```
Resolve-DnsName -Name testhost.testing.local -DnsSecOK
```

> **Note:** The -DnsSecOK switch in the prior command sets a bit flag in the query to the DNS server that indicates we want the RRSIG signature data too.

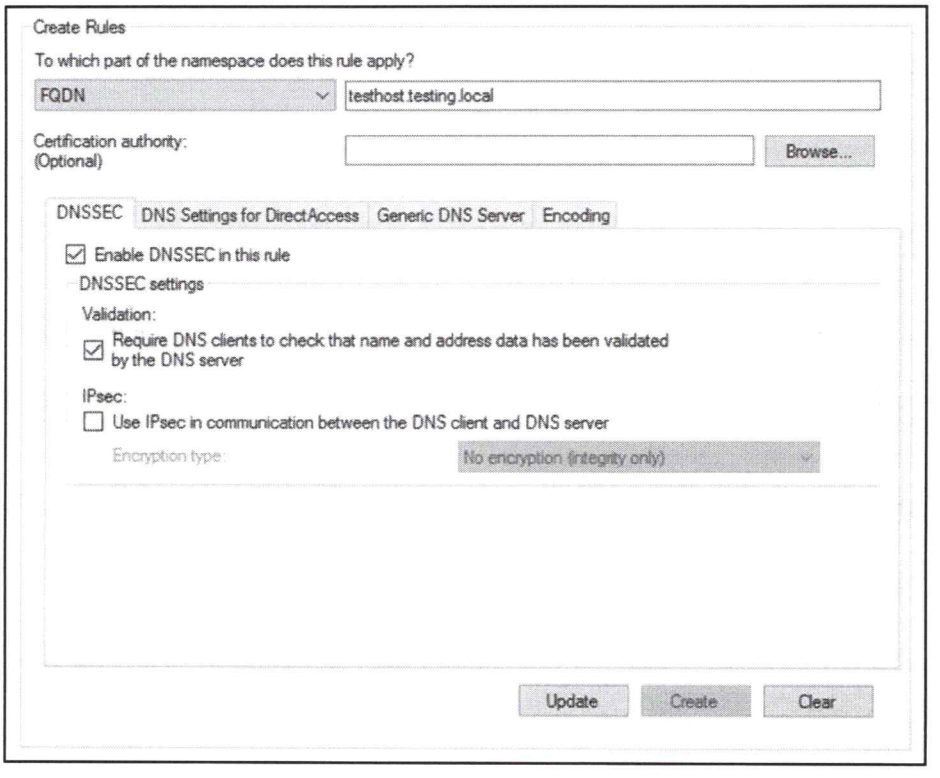
Configure DNSSEC Clients

View your current Name Resolution Policy Table (it may be empty or blank):

```
Get-DnsClientNrptPolicy -Effective
```

Open the Group Policy Management console (GPMC) from the Tools menu in Server Manager and edit the "Default Domain Policy" GPO at the top of your domain.

In the Default Domain Policy GPO for your domain, navigate to Computer Configuration > Policies > Windows Settings > Name Resolution Policy.



On the right-hand side, the GPO is showing the interface for adding name resolution policy rules. In that interface, go to the DNSSEC tab and set these options:

- Pull-down menu to select the part of namespace = FQDN
- Text box next to pull-down menu = testhost.testing.local
- Checkbox named "Enable DNSSEC in this rule" = Checked
- Checkbox named "Require DNS clients to check…" = Checked
- Checkbox named "Use IPSec…" = **Not** Checked

Click the Create button and notice that a new NRPT rule has been added at the bottom.

At the very bottom, click the Apply button (you may need to scroll down in the GPO).

> **Note:** When you click Apply, it might appear that nothing has happened, but the changes have been saved to the GPO. The Apply button does not grey out.

Refresh Group Policy and wait a few moments:

```
gpupdate.exe /force
```

View your current NRPT and you should now see an entry for testhost.testing.local:

```
Get-DnsClientNrptPolicy -Effective
```

Notice in the output from the prior command that "DnsSecValidationRequired : True" means that the client is checking for a flag in the DNS response from the DNS server confirming that the DNS server validated the DNSSEC signature in the name resolved. This flag bit in the response to the client, however, can only be protected with IPSec.

Confirm DNSSEC is working by viewing the RRSIG record for testhost.testing.local:

```
Resolve-DnsName -Name testhost.testing.local -Type RRSIG
```

Finished Already?

View your trust anchor signing keys and DNSSEC settings:

```
Get-DnsServerTrustAnchor -Name testing.local | Format-List *

Get-DnsServerDnsSecZoneSetting -ZoneName testing.local
```

Test your DNSSEC configuration for problems:

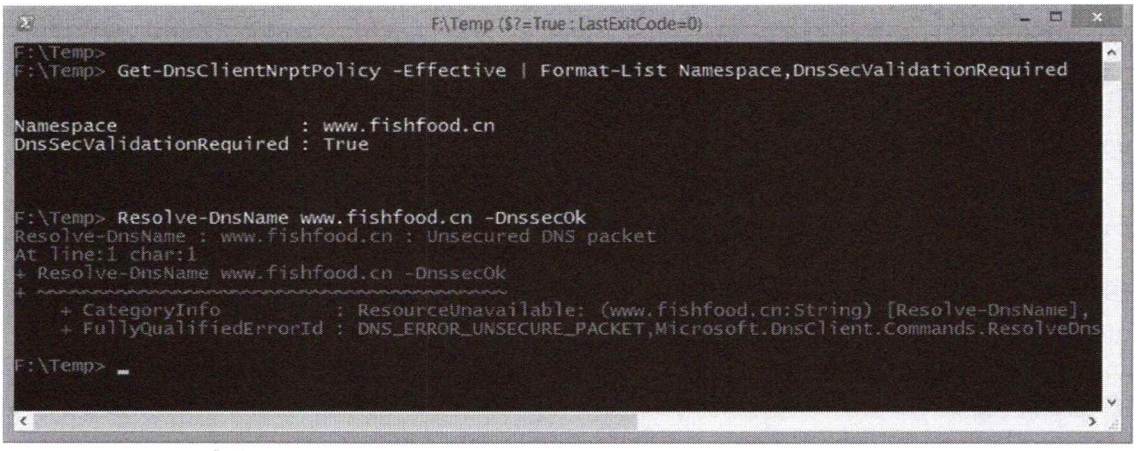
```
Test-DnsServerDnsSecZoneSetting -ZoneName testing.local
```

What about seeing an DNSSEC authentication failure? This is more difficult to demonstrate on one VM with no Internet access. This is because when directly querying an authoritative DNS server for a particular zone, such as your own DNS server for the testing.local zone right now, the response is *always* considered valid.

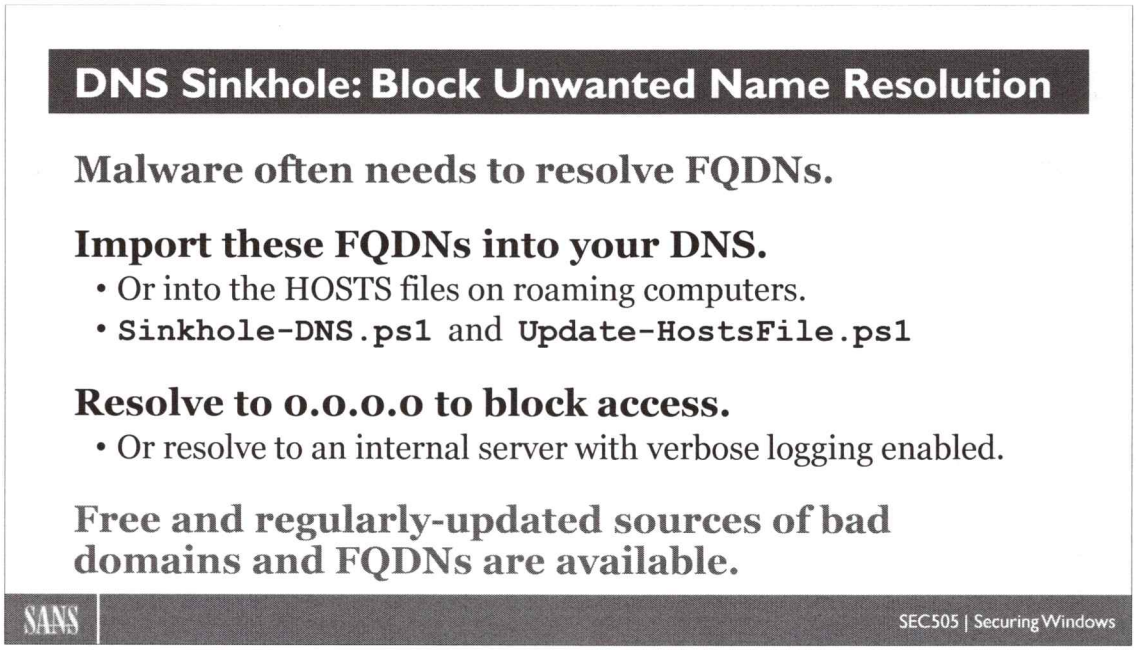
If you want to see an DNSSEC authentication failure when you get back home, you will need a computer with Internet access, and then run the following command to require DNSSEC validation for a FQDN that you happen to know does not support DNSSEC, such as, for example, www.fishfood.cn in China:

```
Add-DnsClientNrptRule -DnsSecEnable -DnsSecValidationRequired
  -Namespace "www.fishfood.cn"
```

When DNSSEC authentication fails for a FQDN that requires it, the error message will report an "Unsecured DNS packet" and Windows will refuse to resolve the IP address even if one was returned by the DNS server. The following screenshot shows what the error message looks like.



Hopefully the fishfood.cn zone will stay unsigned for a while…



DNS Sinkhole: Block Unwanted Name Resolution

Many forms of malware perform DNS name resolution as part of their operations. A browser infection, for example, might start with a command to download a spyware binary, but the server's IP address isn't hard-coded into the infection code, the code uses a fully-qualified domain name (FQDN) and DNS so that the download server(s) can be more easily changed by the attacker. You might not be able to block the initial browser infection, but you could block the spyware download by editing your DNS servers or hosts files to include the wrong IP address for the spyware download server.

Infected Host Detection

Furthermore, if the "wrong" IP address you specify in DNS for the undesired FQDN actually points towards an internal HTTP/FTP/IRC/SMTP server of your own, you would be able to log 1) the URL of the files being requested, the IRC commands entered, the messages being sent, or other protocol interactions, and 2) you'd see the IP address of your own infected computer making the request. By examining the logs of this honeypot-like server you could extract a list of infected machines to clean and better analyze the malware involved.

Hosts File on Laptops and Tablets

The textual hosts file on clients could also be edited by script to thwart malware name resolution (on Windows, it's located at %SystemRoot%\System32\Drivers\Etc\hosts). This is especially useful on laptops which roam outside the office and use other DNS servers not under your control. A nice way to set this up is to use a DFS share point or internal web server to host your master hosts file, then implement a boot-up or scheduled script on your laptops to periodically replace their local hosts files with the central master file. When a new malware specimen runs rampant, you would then configure DNS and

the master hosts file with the necessary FQDNs to point towards your internal honeypot-like server. Because malware often edits the hosts file itself for its own purposes, frequently overwriting that file with a good version helps impede that malware too.

Lists of Bad FQDNs

Where can we find out about the FQDNs being resolved by widespread malware? One source are the anti-virus vendors in their write-ups of new variants, and your AV vendor might simply make FQDN blocklists available to subscribers for free, especially if that vendor also sells anti-spyware. You don't have to try to follow every last mutation, just blocking the dominant players is beneficial too. You can also visit web sites like the following which maintain updated lists of undesirable FQDNs for free (or cheap):

- http://www.malwaredomains.com
- http://www.malwaredomainlist.com
- http://www.malwareurl.com
- http://someonewhocares.org/hosts/
- http://www.mvps.org/winhelp2002/hosts.htm
- http://www.urlblacklist.com

Management Scripts

There are a variety of free tools and tutorials for managing blocked domains on BIND and Windows DNS servers, just do an Internet search on terms like "dns sinkhole tool malware blocking" and similar. Here are a few links to get started:

- http://www.malwaredomains.com/bhdns.html
- http://www.sans.org/windows-security (look for DNS sinkhole article)
- http://en.wikipedia.org/wiki/DNSBL
- http://www.sans.org/reading_room/whitepapers/dns/dns-sinkhole_33523

On your USB/CD you will find a PowerShell script named Sinkhole-DNS.ps1 for Windows DNS servers running on Server 2008 or later. This can import text files of bad FQDNs and domain names into local or remote DNS servers and set the sinkhole IP address that should be returned when there are queries for one of the bad names.

On your USB/CD is another script, Update-HostsFile.ps1, for roaming computers.

As mentioned before, you also want to block all DNS traffic at the perimeter except to/from your own authorized DNS servers. The TDSS rootkit (a.k.a., Alureon and TDL4) can spread by spoofing DHCP replies in order to configure victims with a DNS server under the control of the attacker; the attacker's DNS server can then redirect victims to web pages with malware-installing exploits. The TDSS rootkit infects 64-bit systems too (http://www.theregister.co.uk/2011/06/03/tdss_self_propagation_powers/).

DNS Policies (Server 2016 and Later)

Server 2016 introduced "DNS Policies" to implement custom DNS response rules based on a variety of criteria; for example, DNS Policies can be used to implement two

different views of a zone depending on whether the client is inside or outside the LAN (this is often called "split brain" or "split horizon" DNS).

Note that these DNS Policies are not visible in the graphical DNS snap-in management tool. DNS Policies are managed with PowerShell, such as these cmdlets:

```
Get-DnsServerQueryResolutionPolicy

Add-DnsServerQueryResolutionPolicy

Remove-DnsServerQueryResolutionPolicy
```

And if you'd like to use DNS Policies for split-brain DNS, then see the help for these too:

```
Add-DnsServerZoneScope

Add-DnsServerResourceRecord -ZoneScope

Add-DnsServerClientSubnet
```

DNS Policies can also be used for sinkholing unwanted names. Unfortunately, though, a DNS Policy rule for a sinkholded name only permits either 1) rejecting the query with an error response or 2) ignoring the query entirely. This means that we cannot use DNS Policies to return an IP address of our choice for sinkholed name, such as to an internal server for logging and network forensics. And, again, these sinkholed names will not be visible in the DNS snap-in.

If you wish to use DNS Policies for sinkholing anyway, here is an example of sinkholing "www.fishfood.cn" by making the DNS server simply ignore any queries for that name:

```
Add-DnsServerQueryResolutionPolicy -Name "SinkholePolicy"
  -Action Ignore -FQDN "EQ,www.fishfood.cn"
```

If you'd prefer the DNS server to respond with an error (SERV_FAIL) in response to queries for sinkholed names, change the -Action argument above to "Deny."

On Your Computer

This lab has multiple parts.

Hosts File

In PowerShell, please switch to the C:\SANS\Day6-Servers\DNS folder:

```
cd c:\SANS\Day6-Servers\DNS
```

Review the help text embedded inside the Update-HostsFile.ps1 script:

```
get-help -full .\Update-HostsFile.ps1
```

Update the hosts file with a list of bad FQDNs that we don't want to resolve correctly, then view the contents of the hosts file (and close Notepad afterwards):

```
.\Update-HostsFile.ps1 -FilePathOrURL .\TestNamesToSinkhole.txt

.\Update-HostsFile.ps1 -edithostsfile
```

When you are using a logging/honeypot server, then the sinkhole destination IP can be changed from 0.0.0.0 to your own server for logging or a "Web Site Restricted" page:

```
.\Update-HostsFile.ps1 -FilePathOrURL .\TestNamesToSinkhole.txt
  -SinkholeIP 10.1.1.1
```

```
.\Update-HostsFile.ps1 -EditHostsFile
```

Reset the hosts file back to the factory default:

```
.\Update-HostsFile.ps1 -ResetToDefaultHostsFile
```

> **Note:** If you wish, on your host computer (not in the VM), if you have Internet access, you can run the Update-HostsFile.ps1 script with no arguments and the script will reach out to www.MalwareDomains.com and download an updated list of thousands of bad FQDNs, which will be used to update your hosts file.

DNS Servers

There are many functions available for managing DNS (PowerShell 4.0 and later):

```
get-command -module DnsServer
```

Display global configuration settings for a local or remote DNS server:

```
Get-DnsServer
```

Display DNSSEC information for the testing.local DNS zone:

```
Get-DnsServerDnsSecZoneSetting -ZoneName "testing.local"
```

Display the DNS server's forwarding settings, e.g., the IP addresses of other DNS servers:

```
Get-DnsServerForwarder
```

Assign two IP addresses to be used as DNS forwarder target servers:

```
Get-DnsServerForwarder -IpAddress
  @("208.67.222.222","208.67.220.220")
```

Create a new DNS record (an A record) in the testing.local zone:

```
Add-DnsServerResourceRecordA -Ipv4address "10.1.1.4" -Name "VPN"
  -Zonename "testing.local"
```

Display the new DNS record just created:

```
Get-DnsServerResourceRecord -Name "VPN" -RRType "A"
  -Zonename "testing.local"
```

DNS Sinkhole

See the formatted help from the Sinkhole-DNS.ps1 script:

```
get-help -full .\Sinkhole-DNS.ps1
```

Import a list of bad FQDNs into your DNS server, resolving evil names to 10.1.1.1:

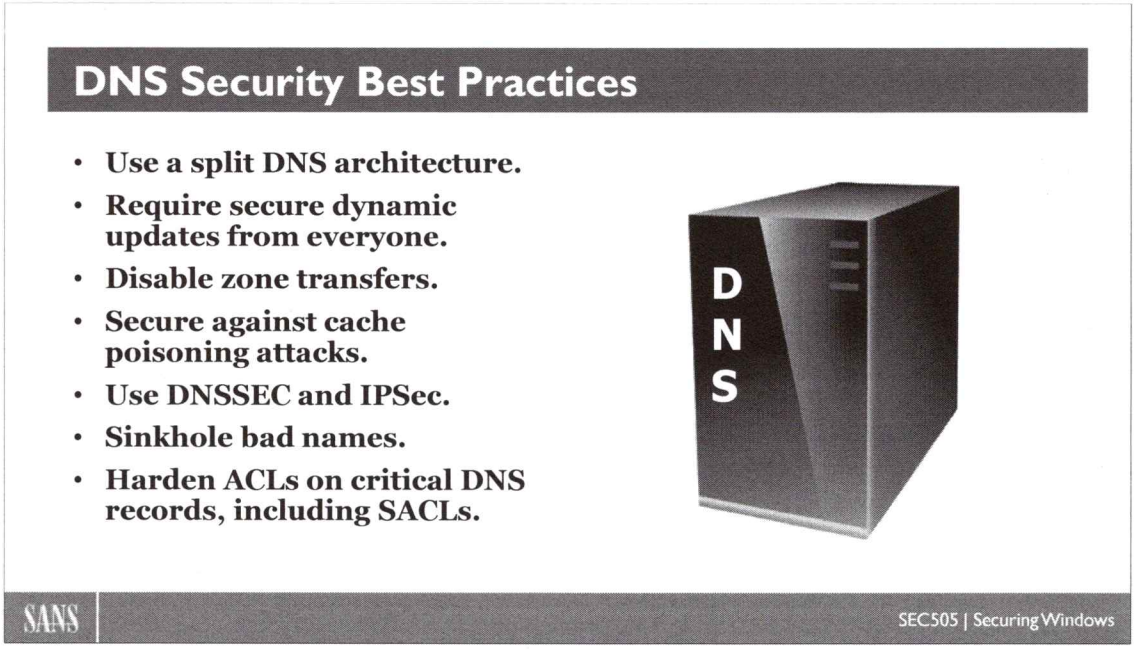
```
.\Sinkhole-DNS.ps1 -InputFile .\TestNamesToSinkhole.txt
  -SinkholeIP 10.1.1.1
```

Now, in the graphical DNS management tool, do a refresh at the server level to see the changes. (No matter how many FQDNs are sinkholed, only one small zone file is used, and none of these records are replicated through Active Directory.)

When you delete the records for the bad FQDNs, no other DNS records are touched:

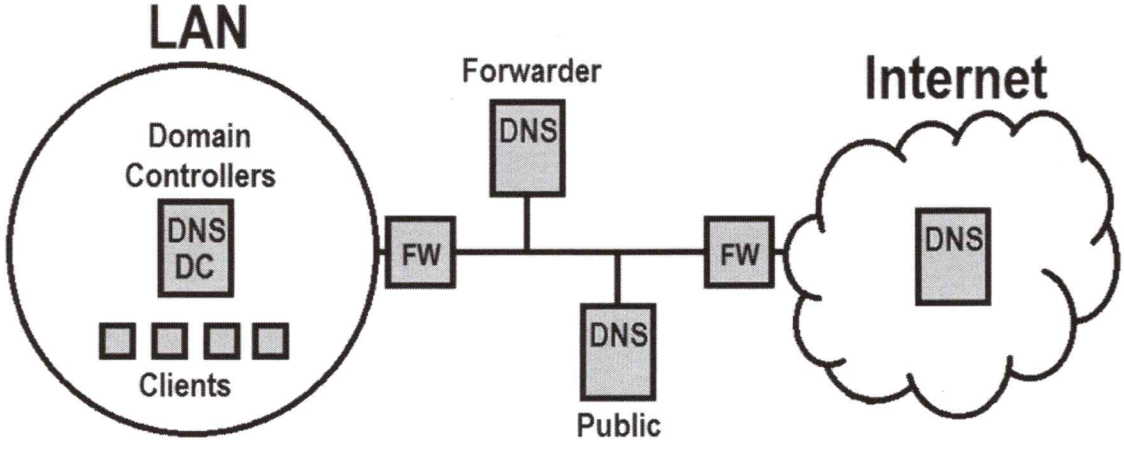
```
.\Sinkhole-DNS.ps1 -DeleteSinkHoleDomains
```

(If you refresh the graphical DNS management tool, only the bad records are gone now.)



DNS Security Best Practices

The following items summarize the security best practices for DNS. We won't talk about them all here now, but it's good to have the list for reference because of the importance (and vulnerabilities) of DNS.



In the diagram above, the firewall (FW devices) blocks all DNS traffic between the LAN internal DNS servers, which are also domain controllers, and all the thousands of DNS servers scattered around the Internet. The firewall only allows DNS traffic between the internal DNS servers and the forwarder DNS servers in the DMZ. Forwarder DNS servers are permitted to query other DNS servers on the Internet around the world. Internal LAN clients may only submit their queries to their local LAN DNS servers. When another Internet DNS server needs to query the public records of the organization,

the queries are resolved by the public DNS servers in the DMZ, not the internal LAN DNS servers. This arrangement of DNS servers and firewall rules is called a "split DNS" architecture.

External DNS (Forwarders and Public DNS Servers in the DMZ)

- Block all access from the Internet to your internal DNS servers. Allow Internet access only to your external DNS servers which are located on a perimeter segment of your firewall, such as in the DMZ.

- If you are hosting authoritative records for your own DNS domains and these records must be Internet-accessible, the DNS servers on the perimeter segment hosting these records should be used only for this one purpose, hence, they should not also be caching DNS servers which accept recursive queries. On these authoritative DNS servers, disable support for recursion (Advanced tab), delete any configured forwarders (Forwarders tab), and uncheck the option to use root hints too (Forwarders tab). These authoritative DNS servers should only have the bare minimum records necessary for your SMTP, VPN, HTTP and other public servers (assume that all DNS records on these servers will eventually be extracted, even if zone transfers are disabled).

- Strongly consider using DNSSEC on external servers.

- On the firewall's perimeter segment or DMZ, install two or more caching DNS servers which accept recursive queries. These servers should not be authoritative for any DNS records. Only accept queries from the internal LAN (Interfaces tab).

- Configure internal DNS servers to forward their queries to the external caching/recursive DNS servers for the resolution of FQDNs on the Internet. Don't allow internal DNS servers to directly query other DNS servers around the world.

- Have more than one external and internal DNS server for load balancing and fault tolerance. DNS is particularly vulnerable to DoS attacks, especially DNS servers which accept recursive queries. Avoid connecting all DNS servers to the same segment, switch or router since this creates a single point of failure.

- Be cautious of which DNS servers you choose to forward queries to if you don't control them. Do you trust those DNS servers? Are they hardened and monitored boxes? Consider doing your own iterative queries if you can't trust the DNS servers of your ISP or other DNS hosting providers, e.g., www.opendns.com.

- Use the "Secure cache against pollution" option on the Advanced tab of the property sheet of the DNS server if it is exposed to the Internet. Internal-only DNS servers can leave this box unchecked, but only if they only query other internal DNS servers. Best to enable the option on all DNS servers however.

- Disable zone transfers on both internal and external DNS servers. If it is required, only permit zone transfers to selected IP addresses.

- Update the root hints information on your DNS servers performing iterative queries and periodically check that they are current and unmodified.

Internal DNS (DNS Servers/Domain Controllers Inside the LAN)

- Use Active Directory-integrated zones on internal DNS servers, but not on external Internet-accessible servers.

- Require secure dynamic updates on internal DNS servers. Disable dynamic updates entirely on external or Internet-accessible DNS servers.

- The DNS records of critical systems (such as mail servers, domain controllers, web servers, etc.) should be protected by assigning them restrictive AD permissions so that only Administrators can edit them. Don't forget to enable AD auditing of changes to these records too.

- Require DNSSEC and/or IPSec authentication for all intranet DNS traffic to confirm that clients are connecting to the real DNS servers. Especially require IPSec authentication if you use zone transfers since zone transfers are not protected by DNSSEC.

- Configure DHCP dynamic update credentials, especially when the DHCP service is hosted on a domain controller.

- Use DNS sinkhole techniques to block the resolution of unwanted names, such as the names resolved by malware, especially when one's proxy server or web content filtering system does not support this feature. Use the same technique with the hosts file on traveling laptops and tablets; when they are outside the LAN, they are likely not using your DNS or proxy servers anymore. With a bit of simple scripting and scheduled jobs, the sinkhole list can be updated daily.

- Enable debug logging when bugs or attacks are suspected. In paranoid environments, the logging can be enabled continuously as long as the performance penalty and drive space requirements are satisfactory.

- Enable the "Netmask ordering" option on the Advanced tab of the property sheet of the DNS server to slightly optimize network traffic. This is not a security issue.

- After verifying that all services and applications will not fail, try to uninstall WINS and disable NetBIOS (*try*, because you may have to re-enable it again if something breaks). Disable NetBIOS in any case on servers which are directly

accessible from the Internet. Block all incoming and outgoing NetBIOS traffic at the perimeter firewall.

- And, of course, apply the latest Service Packs and patches.

Miscellaneous

- Examine the hosts and lmhosts files on computers suspected of being compromised. Malware often modifies these files.

- Don't forget to save the contents of the DNS hostname cache on machines before they are imaged for forensics purposes: "ipconfig.exe /displaydns > a:\file.txt".

- Use Performance Monitor or other scripts to track the accessibility and health of DNS servers. If DNS has problems, Active Directory and everything else has problems too. DNS is the forgotten Achilles Heel for many network services.

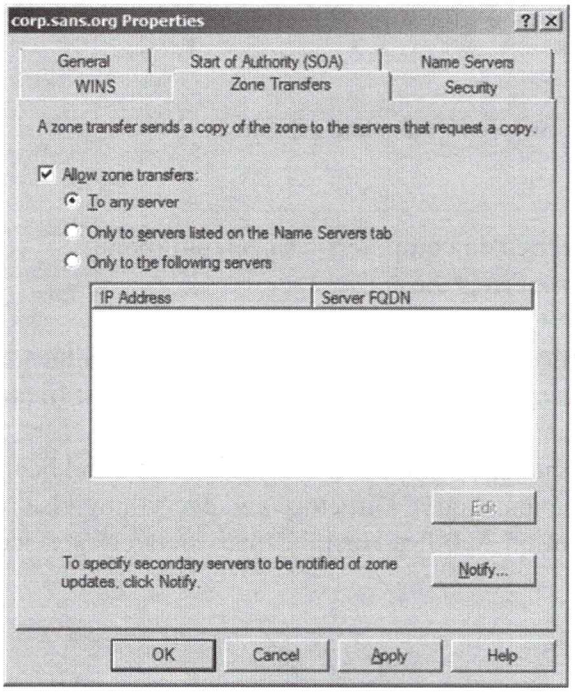
How To Disable Zone Transfers

AD-integrated DNS servers can still act as primaries for regular secondary DNS servers. Remote users, including hackers, can still use the NSLOOKUP.EXE utility to list all records in a domain too (ls –d *targetdomain.com.*). To control which secondaries and NSLOOKUP users can obtain resource records, the property sheet of each domain in the DNS snap-in includes a Zone Transfers tab.

You can disable zone transfers and NSLOOKUP listings entirely by unchecking the box labeled "Allow Zone Transfers". The Name Servers tab will list all DNS servers which are authoritative for the zone, including other AD-integrated DNS servers and non-AD secondary servers. The Notify button is used when the DNS server is acting as a primary to a non-AD secondary DNS server.

Try It Now!

To disable zone transfers entirely, right-click the desired zone > Properties > Zone Transfers tab > uncheck the box labeled "Allow zone transfers". It may already be disabled by default.



Controlling zone transfers is important because hackers will attempt to download all records from your DNS servers in order to discover which IP addresses are in use and to better guess what is hosted on each server (hostnames typically indicate the major service, e.g., www, ftp, pdc). Hackers can find out the IP addresses of the two official Internic-registered DNS servers for your domain(s) by doing a whois lookup at http://www.networksolutions.com. To test whether your DNS servers are promiscuously revealing zone data, use NSLOOKUP with the listing command, and don't forget the period at the end of the domain name ("ls -d *domainname.com.*").

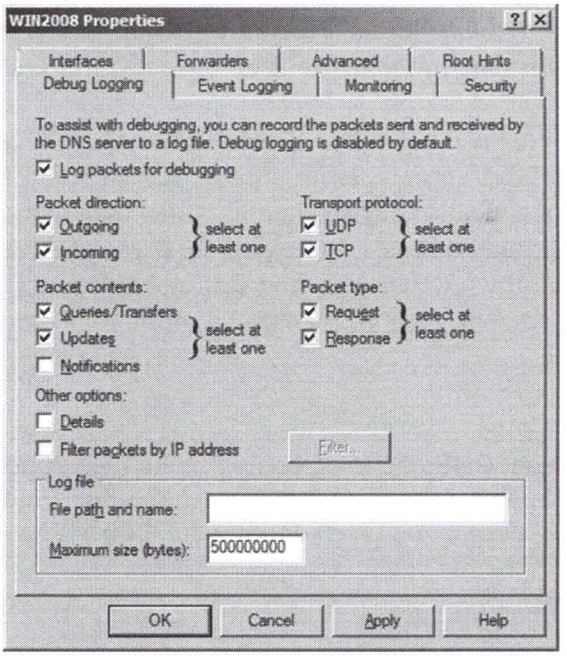
If you are using AD-integrated DNS zones (which is recommended) and no non-AD secondaries, then disable zone transfers entirely.

If secondaries or stub zones are in use, then permit zone transfers only to the other servers on the Name Servers tab or control by source IP address. When another DNS server hosts a stub zone with the NS records of one's own zone, that other DNS server will periodically pull your NS records using a zone transfer on TCP/53, hence, you'll need to permit that other DNS server's IP address to request zone transfers.

How To Enable DNS Logging

DNS activities can be logged to a text file for troubleshooting or security analysis.

> ***Try It Now!***
> To enable DNS activity logging, go to the Properties of the DNS server in the DNS snap-in > Debug Logging tab > check the desired activities to be logged > OK.



The DNS activities that can be logged include the following:

- Query -- queries received from clients.
- Notify -- notification messages received from other DNS servers.
- Update -- dynamic updates received from other computers.
- Questions -- the question section for each DNS query message processed.
- Answers -- the answer section for each DNS query message processed.
- Send -- number of DNS query messages sent.
- Receive -- number of DNS query messages received.
- UDP -- number of DNS requests received over a UDP port.
- TCP -- number of DNS requests received over a TCP port.
- Full Packets -- number of full packets written and sent by DNS.
- Write Through -- number of packets written through and back to the zone.

The default location for the DNS log is \%SystemRoot%\System32\dns\Dns.log. You can change the folder and filename, as well as the size. Once the log file reaches its maximum size, logging will wrap around and start writing at the top of the file.

Secure Cache Against Pollution

DNS cache poisoning is a type of attack in which bogus query responses are sent to a DNS server in an attempt to overload and/or "pollute" the server's cache of DNS records. For example, a hacker might pollute a DNS server's cache with the incorrect IP addresses for popular websites.

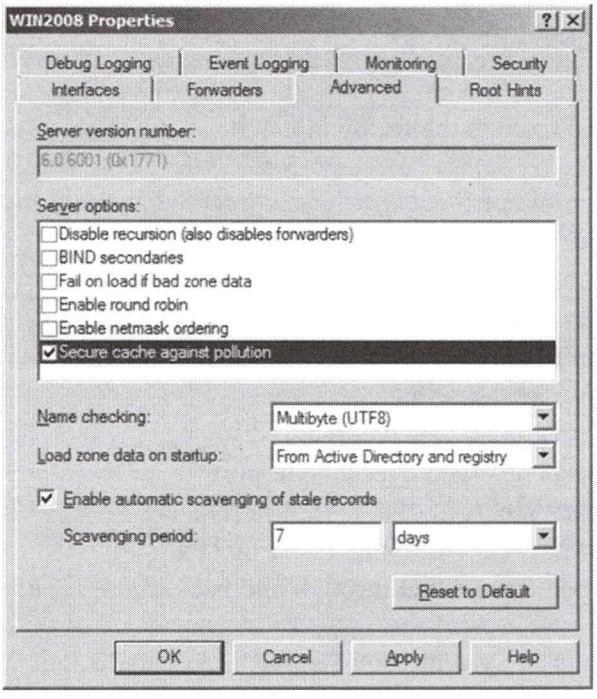
Any unsolicited DNS replies your server receives are automatically dropped, no configuration required (KB241352). This prevents a certain type of DNS cache

poisoning where an attacker sends DNS replies with incorrect information, even though the DNS server never sent a query for this data in the first place.

Another anti-poisoning option must be enabled manually. If the "Secure Cache Against Pollution" option is enabled, when your DNS server queries another server, and that other DNS server returns additional records outside of the DNS domain of the data requested, then your DNS server will drop the additional records. For example, if your DNS server queries another server for the IP address of www.sans.org, and that other DNS server's reply includes records for www.microsoft.com, then your DNS will not include the www.microsoft.com records in its cache.

Try It Now!

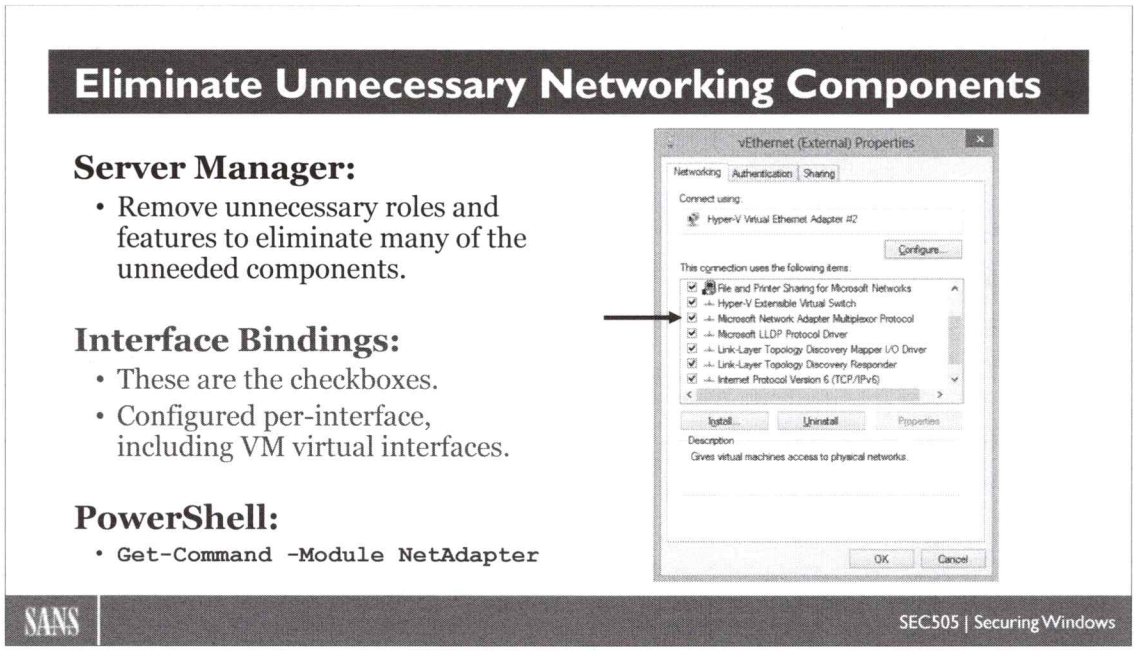
To enable partial protection against DNS cache poisoning, go to the Properties of your DNS server in the DNS snap-in > Advanced tab > check the box "Secure Cache Against Pollution". It may already be checked by default on your system.



The "BIND Secondaries" option is when there are older versions of BIND (e.g., 4.9.4) being used as secondaries. These older versions use a different format for zone transfers than newer versions. Windows DNS uses the newer fast zone transfer BIND format by default.

"Netmask Ordering" will arrange the answers in the DNS server's responses such that the IP address(es) of the answers most similar to the clients will be placed near the top of the answers list. Clients typically use the answers in order from top to bottom. This helps to optimize the overall performance of the network because clients tend to connect to servers closest to them on the network.

The "Name Checking" list permits the control over hostname syntax and character set verification. Hostnames can be required to be RFC 1123 compliant or not, and UTF-8 Unicode characters are also supported for international character sets.



Eliminate Unnecessary Networking Components

Your protocol stack is on the very front lines of battle between you and your adversaries. If your firewall exposes various ports to the Internet, you are allowing your adversaries to talk directly to your protocol stack. A "binding" is an internal path of communication between a network adapter card and a protocol, or between a protocol and a service. A "networking component" is anything which can be bound to a NIC. Uninstall unnecessary networking-related roles and features with Server Manager, and unbind any remaining unnecessary components from the NICs, especially those interfaces which are exposed to the Internet.

> ***Try It Now!***
> To unbind a networking component, open "Network and Sharing Center" in Control Panel > Change Adapter Settings > right-click on the desired NIC > Properties > uncheck each box represents the unwanted binding.

At a minimum, you should disable both the "File and Printer Sharing" (Server) and "Client for Microsoft Networks" (Workstation) services from your Internet-attached NIC by unchecking their boxes, though you might keep these bindings for a NIC facing the internal LAN. Unbind as many other protocols/services as possible without breaking your desired functionality.

Managing bindings is a way to reach a compromise when you must run a dangerous service or protocol. For example, if you have two network adapter cards and you must run the Server service, then unbind "File and Printer Sharing" from you Internet-attached network card. By removing the binding, you prevent packets from the Internet from reaching the Server service even though that service is installed and running. Don't connect one NIC directly to the internal network and expose the other the Internet. If the

box is taken over, you don't want to permit the attacker to turn it into a router. All interfaces should be connected to one or more service/DMZ segments off of the firewall.

If you will not be using SMB to manage your server at all, then you can uninstall "File and Printer Sharing" entirely by highlighting the service and clicking Uninstall. Be prepared to be inconvenienced later, though, and test to see if any of your management tools fail because of this (if one does, then simply add the service back).

PowerShell

With PowerShell 4.0 and later, you can display the names of your adapter:

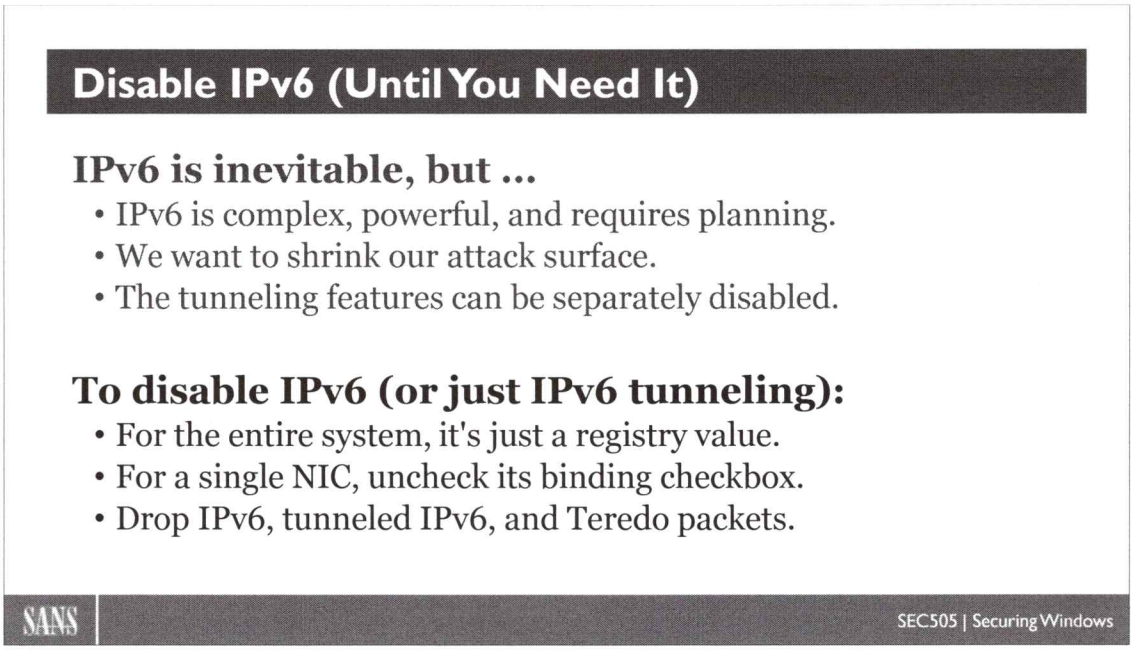
```
Get-NetAdapter
```

Once you have the name of the adapter, such as "Ethernet", then display its bindings:

Get-NetAdapter -Name "Ethernet" |

DNS Name Resolution Only

As discussed in a different part of this course, we want to only use DNS name resolution, not NetBIOS or Link-Local Multicast Name Resolution (LLMNR). Because of DNS insecurities, we also want to only query internal DNS servers using our split DNS architecture, require DNSSEC when possible, sign DNS query traffic with IPSec, use either secure dynamic updates or static DNS records, and regularly update "sinkholed" names on the internal DNS servers too. Ideally, a server would not be a client to any DNS server in order to eliminate this exploitable dependency, but this is extremely difficult to do in most circumstances.



Disable IPv6 (Until You Need It)

Your use of Internet Protocol version 6 (IPv6) is inevitable in the long run, but until your organization is ready to use and manage it carefully, it should be disabled.

So, the recommendation is not to disable in every circumstance, the recommendation is to disable it until 1) you need it, and 2) your firewalls, routers, switches, IDS/IPS sensors, wireless Access Points, servers, workstations, laptops, tablets, smart phones, printers, and (especially) your IT staff are ready to handle it.

Your organization's own "IPv6 Day" may come next year, in five years, or in ten years. But until that day comes, IPv6 is a risk because it is complex, powerful and built into the protocol stack, i.e., it is near the "front lines" between your machine and your attackers.

Why Not Disable IPv6?

What are some reasons why IPv6 should not be disabled? These are mostly valid issues, but the question is whether they are decisive. These are listed from most to least convincing:

- IPv6 is inevitable. (This is true, and by far the best argument.)
- Microsoft DirectAccess sometimes requires IPv6, especially older versions. Are you using or planning to use Microsoft DirectAccess?
- Microsoft doesn't want IPv6 disabled, they do not test their patches on systems with IPv6 disabled, and future applications or features may require IPv6.

- There are some known negative issues when IPv6 is disabled, but these are rather rare (KB816103, KB977623) except on Small Business Server (SBS). On SBS, disabling IPv6 caused big problems, especially related to RRAS and Exchange, but SBS is a discontinued product.

- There are IPv6 dependencies with Remote Assistance, HomeGroup, Windows Mail, and some Windows-only P2P applications, but it depends on the details. Besides, what companies actually use HomeGroup, Windows Mail or Windows-only P2P applications? Remote Assistance does not always require IPv6, it works fine with IPv4, and the issues are minor anyway.

- "IPv6 was subjected to a rigorous testing process and security review," said a Microsoft engineer once, "before we allowed Windows Vista and Windows Server 2008 to ship with it enabled and preferred." Yah, that's what they said about Internet Explorer too. This is basically the "*don't worry you can trust us*" argument...

- Teredo, 6to4, ISATAP and other transition technologies don't work without IPv6. But this is a circular argument, these technologies aren't needed until we've decided that we need IPv6 to begin with.

Like any complex technology, if you need it, great! Deploy and manage it safely. But the overarching security principle here hasn't been revoked: you should shrink your attack surface by disabling or restricting what you don't need (for example, MS13-065 was an ICMPv6 Ping of Death vulnerability discovered in 2013). If you won't be using IPv6 in the next few years, get rid of it. Later, when you do need it, turn it back on, but manage it carefully and correctly.

How To Disable IPv6 (Entire Computer)

KB929852 describes the registry value to modify to disable IPv6 components on Windows Vista, Server 2008, and later systems. Windows XP and Server 2003 do not include IPv6 by default.

Create or edit the 32-bit DWORD value named "DisabledComponents" located at HKLM\SYSTEM\CurrentControlSet\Services\Tcpip6\Parameters\. Set that value to "0xff" to disable all IPv6 components on all adapters, except for the IPv6 loopback adapter, which cannot be disabled.

How To Keep IPv6, But Disable IPv6 Tunneling (Entire Computer)

Refer to KB929852 for instructions on how to disable IPv6 only for certain types of interfaces, such as tunnel interfaces. Disabling just tunnel interfaces (ISATAP, 6to4, Teredo) would be a good option for organizations which do not want to eliminate IPv6 entirely, but are worried about IPv6 evasion of network security devices. Alternatively, if you disable IPv6 completely today, then, when you later re-enable IPv6 a few years from now, you could first enable IPv6 excluding the tunnel interfaces, then possibly enable the tunnel interfaces too after that. This might provide a gentler migration path anyway.

This registry change can be incorporated into the binary image you use to build new machines, set with a Group Policy startup script, set with a custom ADM/ADMX template in Group Policy, or set remotely with a script.

On your course CD you have an ADMX template for Group Policy control over IPv6 and its tunneling features. To use the ADMX template, you must copy it and its associated ADML file into the correct directories on the computer where the Group Policy Management console (GPMC) will be used to edit any GPOs.

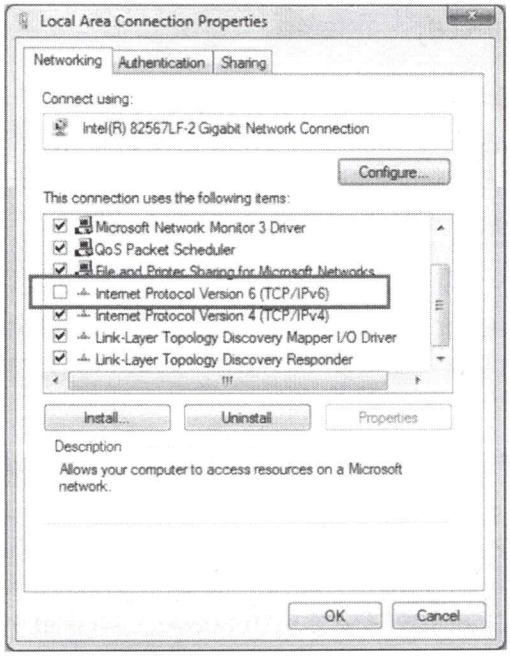
```
cd C:\SANS\Day6-Servers\IPv6

copy .\IPv6Configuration.admx $env:WinDir\PolicyDefinitions

copy .\IPv6Configuration.adml $env:WinDir\PolicyDefinitions\en-US
```

> **Note:** If your language locale is not "en-US", copy the ADML file into the appropriate folder instead. It will be in the PolicyDefinitions directory too.

After copying the ADMX and ADML files, open a GPO and navigate to Computer Configuration > Policies > Administrative Templates > Network > IPv6 Configuration. Here you will find the IPv6 policy settings.

How To Unbind IPv6 (Per Interface)

Alternatively, if you wish to unbind IPv6 from just one interface, but leave it enabled on the others, go the properties of that interface and uncheck the box for its IPv6 binding.



Drop IPv6 Traffic: Perimeter and Host-Based Firewalls

Your perimeter firewall and the host-based Windows Firewall on every system should be configured to drop all inbound and outbound packets using IPv6, including:

- EtherType 0x86DD (in the IEEE 802.3 Ethernet header to indicate IPv6 payload)
- IPv6 tunneled over IPv4 (protocol ID number 41 in the IPv4 header)
- UDP port 3544, both source and destination (Teredo)

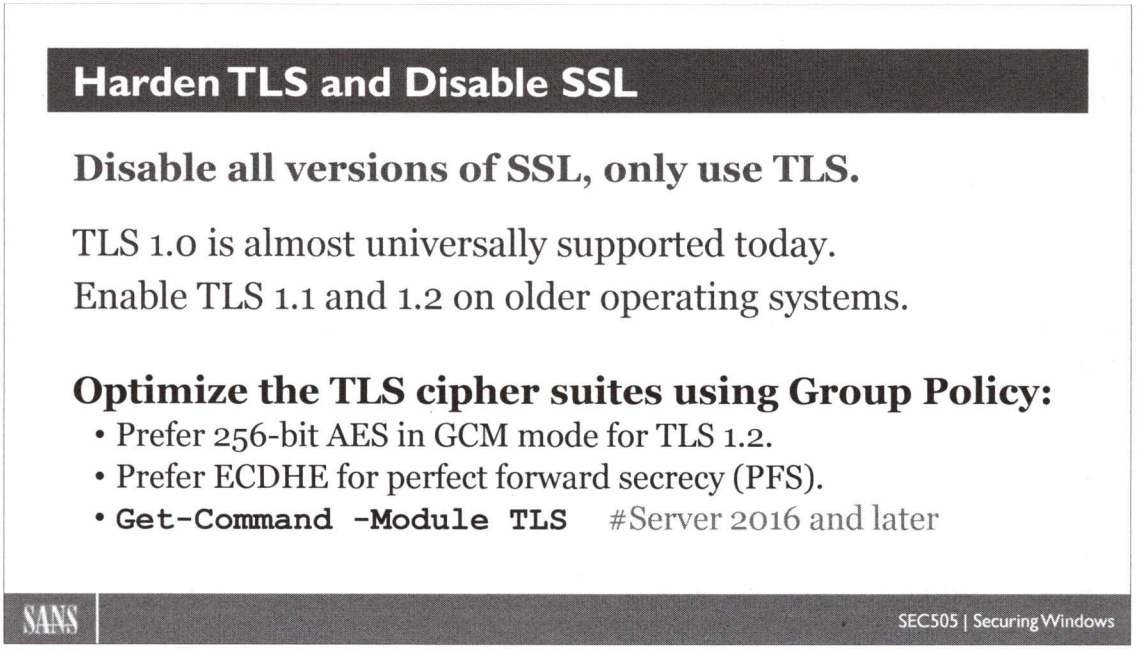
Though it should be unnecessary, here are additional IPv6 headers that you might wish to know about, either to drop or to allow:

- ICMPv6 (protocol ID number 58 in the IPv6 header)
- IPv6 Routing (protocol ID number 43 in the IPv6 header)
- IPv6 Fragment (protocol ID number 44 in the IPv6 header)
- IPv6 No Next Header (protocol ID number 59 in the IPv6 header)
- IPv6 Destination Options (protocol ID number 60 in the IPv6 header)

However, only drop Link-Local Multicast Name Resolution (LLMNR) packets on UDP 5355 at the perimeter firewall, not on host-based firewalls, if you intend to use it internally. LLMNR is not unique to IPv6, it is also used with IPv4.

But We Use IPv6!

If you use IPv6 deliberately, that's great, you are ahead of the curve and where everyone will be in the future. In this case, just manage it carefully so that the benefits outweigh the risks. A good place to start is to review the recommendations in NIST Special Publication 800-119: *Guidelines for the Secure Deployment of IPv6*.



Harden TLS and Disable SSL

To state it mildly, hardening SSL and TLS can be complicated, but it is necessary.

There is the choice of protocol (SSL or TLS), protocol version (TLS 1.0, 1.1, 1.2) and cipher suite (many cipher, key size and hashing options). But we also must worry about browser, operating system and non-browser application compatibility. SSL/TLS options are negotiated between client and server, but client and server can have many options configured. When we control the clients and servers, such as in our own organizations, the choices are easier, but when we do not control the clients or servers, such as the customers visiting our e-commerce web servers or when our own users browse third-party sites, the issues become complicated again. Many clients are not browsers, such as SSL VPN clients or apps on smart phones. And then there is the choice of Certification Authorities (CAs), which CAs to trust, and the details of certificate revocation checking.

And yet we *must* harden SSL/TLS because so much depends on these protocols and there are working exploits against them, such as the LUCKY-13, CRIME and BEAST attacks (KB2643584 and MS12-006).

How To Control Permitted Versions of SSL/TLS

You can enable or disable SSL and TLS separately, including the different versions of these protocols. You can also enable or disable these versions for client and/or server use, e.g., you could forbid TLS 1.0 for IIS, but allow it for Internet Explorer. The Windows module which handles SSL/TLS for both Internet Explorer and IIS is SChannel.dll. SChannel configuration settings are stored in the registry under HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL.

The following sections are exported .REG files which are on your CD. The .REG files can be imported to configure various SSL/TLS options for SChannel. These registry changes can also be pushed out through Group Policy as scripts or .ADMX templates. Please see KB245030 and KB187498 for guidance on the registry keys and values involved, there are too many options available to list and discuss here.

Disable SSL Entirely And Only Use TLS

Unless your testing reveals critical dependencies which cannot be upgraded, the recommendation of this course is to disable all versions of SSL and only use TLS. Most servers and clients today support at least TLS 1.0. At a minimum, disable SSL 1.0 and 2.0 if you must keep SSL 3.0. Fortunately, SSL 1.0 and PCT are disabled by default in XP, Server 2003 and later anyway.

To disable SSL 2.0/3.0 for both client and server use (Disable_SSL_KB245030.reg):

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols]

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 2.0]

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 2.0\Client]
"Enabled"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 2.0\Server]
"Enabled"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 3.0]

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 3.0\Client]
"Enabled"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\
SCHANNEL\Protocols\SSL 3.0\Server]
"Enabled"=dword:00000000
```

See the pattern in the keys? If you wanted to also disable TLS 1.0, create a key named "...\Protocols\TLS 1.0" with subkeys named "Client" and "Server", then set a 32-bit DWORD value named "Enabled" to 0x0 in these Client and Server subkeys.

The value named "DisabledByDefault" doesn't actually enable or disable the protocol permanently, it just sets the default as seen in Internet Explorer and other software.

Another file on the SANS course CD-ROM (Enable_TLS_All_Versions_KB245030.reg) enables all versions of TLS, including TLS 1.0.

These changes are going to require some testing and communication with the help desk, but the goal is to move off SSL at least, then later migrate to TLS 1.1 or better when we can.

Cipher Suite Ordering

On Windows Vista, Server 2008 and later operating systems, it is easy to manage the ciphers, key sizes and hash sizes used by SSL/TLS through Group Policy. The GPO setting is named "SSL Cipher Suite Order" and it is located under GPO > Computer Configuration > Policies > Administrative Templates > Network > SSL Configuration Settings.

The "SSL Cipher Suite Order" setting contains a list of comma-delimited strings indicating ciphers, key sizes and hashing algorithms. The list is ordered from most preferred to least preferred. By removing the items which are unacceptable and rearranging the list, you can control how SSL/TLS settings are negotiated by any software with uses the SChannel security provider, including Internet Explorer and IIS.

The hard part is knowing which cipher suites to select in order to balance performance, compatibility and security.

For example, the following cipher suite ordering prefers later versions of TLS over earlier versions, prefers Elliptic Curve Diffie-Hellman ephemeral (ECDHE) over straight RSA for session key protection, prefers Galois Counter Mode (GCM) over Chaining Block Cipher (CBC) mode, prefers 256-bit session keys over 128-bit keys, and excludes small keys and obsolete algorithms (this file is on the CD in the CipherSuiteOrder.txt file):

```
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P521
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P521
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384_P521
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P521
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P521
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P521
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_RC4_128_SHA
```

Notice that a 128-bit AES key in GCM mode is preferred over a 256-bit AES key in CBC mode; currently, the use of CBC mode is a slightly greater risk than using a 128-bit key. Also, when a suite includes "TLS_ECDHE_RSA" it means that RSA will only be used to authenticate the server, RSA won't be used to protect the symmetric session key. When TLS_ECDHE is used, the public key used to protect the session key is unique for each connection, but TLS_RSA uses the same private key for all connections until the private key is replaced, which may be months or years. If your adversaries are recording your TLS sessions in the hope of later stealing your server's private keys, you'll want ECDHE for the sake of perfect forward secrecy.

Perfect Forward Secrecy (PFS) is a feature of a cryptographic system or protocol where the compromise of one key should not allow the compromise of many other keys used by that system. In this case, we want PFS because the private keys on servers are often used for months or years, and these private keys (and their corresponding public keys) secure the temporary session keys generated by browsers, e-mail programs, and other applications.

The only cipher suite in the above list which is compatible with TLS 1.0 is the very last one, which uses 128-bit RC4, hence, it defeats the BEAST attack and is almost universally supported, but it suffers from being RC4, which is a less-than-ideal cipher. When connecting to your own servers which have TLS 1.2 or later enabled, the better suites will be preferred because they are higher up the list and RC4 will be avoided.

Note: After making any changes to cipher suites, the computer must be rebooted.

As a rule of thumb, never use symmetric keys smaller than 128 bits or RSA public keys smaller than 1024 bits (these are minimums, not recommendations). At the same time, we want to avoid unnecessary performance penalties, so generally avoid symmetric keys larger than 256 bits and RSA public keys larger than 4096 bits. Hence, a 2048 bit RSA public key and a 256-bit AES key would be a good combination for most scenarios. Prefer ECDHE over straight RSA for the sake of perfect forward secrecy. Prefer GCM over CBC. When possible, choose AES to benefit from the hardware acceleration of modern CPUs which have microcode optimized for AES. Finally, while RC4 is universally supported and backward compatible, it's best to migrate to better ciphers like AES.

When new SSL/TLS exploits are published, hopefully you will not be running any of the affected protocols or cipher suites, but, if you are, we now know how to change the settings using the GPO option listed above.

To see the cipher suites offered by your browser, visit:

https://cc.dcsec.uni-hannover.de

To see the cipher suites offered by a public web server, visit:

https://www.ssllabs.com/ssltest/

PowerShell TLS Cmdlets (Server 2016, Windows 10, and Later)

On Windows 10, Server 2016 and later operating systems, you can view and manage cipher suites using PowerShell cmdlets.

To see all TLS-related cmdlets from the TLS module:

```
Get-Command -Module TLS
```

To see the currently-implemented cipher suites in order of preference:

```
Get-TlsCipherSuite | Format-Table Name
```

By the way, the cmdlets for "session ticket keys" are for applications designed to resume a previous TLS session without the server being required to keep session information in memory continuously (see RFC5077). Not relevant here.

Cipher Suite Order for Remote Desktop Protocol (RDP) with TLS

When using Remote Desktop Protocol (RDP) with TLS certificate-based authentication and encryption, which is highly recommended to block MITM attacks, there is a problem with Microsoft's implementation. At the present time, RDP only supports TLS 1.0 with 3DES, RSA and SHA-1. If the cipher suite order configured on the computer does not include an option for this combination, the RDP connection will fail. Hence, for this scenario, you will need to append one more entry to the end of the "SSL Cipher Suite Order" string listed above:

TLS_RSA_WITH_3DES_EDE_CBC_SHA

This cipher suite is not ideal and lacks PFS, but at least it is last on the list of preferred suites. (See the CipherSuiteOrder.txt file on the CD-ROM for all these strings for easy cut-and-pasting.)

Cipher Suite Order (Windows 2000, XP and Server 2003)

Managing cipher suites on Windows 2000, XP and Server 2003 is not as easy. It involves modifying the same SChannel registry key we discussed earlier for disabling SSL, but is not nearly as easy as just pasting a cipher suite ordering string. Because these operating systems are either obsolete or soon will be, let's not discuss the keys here, please refer to KB245030.

Prevent Ignoring Certificate Errors

When an SSL/TLS certificate has expired, isn't from a trusted CA, or has a common name which doesn't match the name of the server being accessed, an error dialog box appears warning the user. However, by default the user is permitted to ignore the error and continue with the connection anyway.

If you want to prevent users from opening an SSL/TLS connection where there is a certificate error, enable the "Prevent Ignoring Certificate Errors" option in Group Policy located in the GPO underneath Computer Configuration > Policies > Administrative Templates > Windows Components > Internet Explorer > Internet Control Panel.

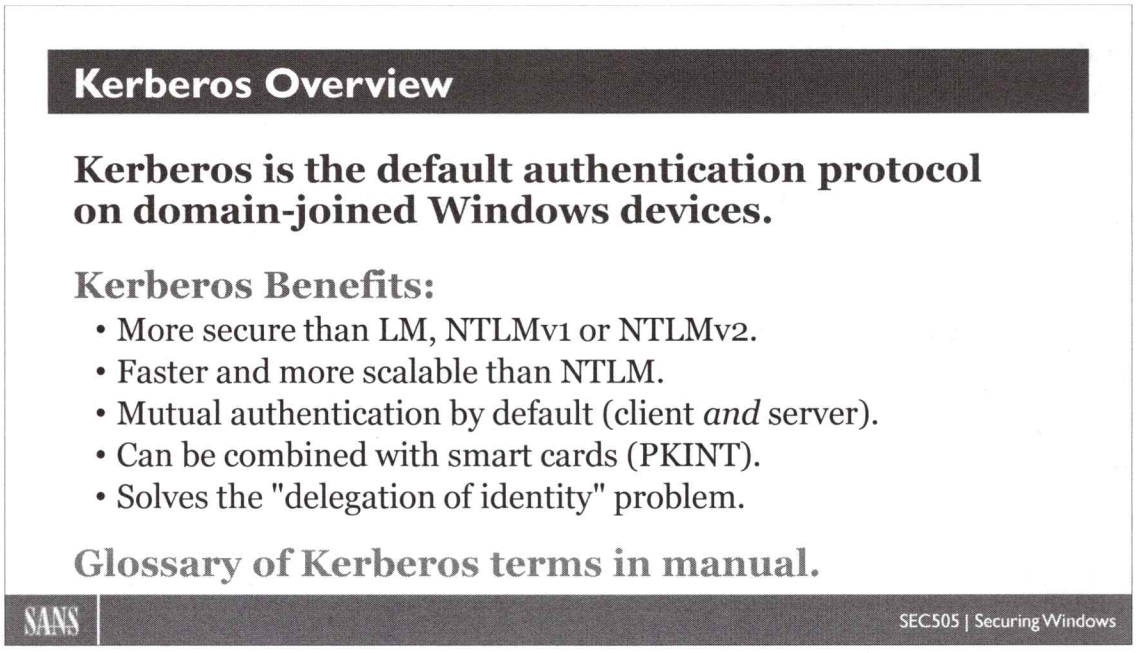
If there are some sites to which you do wish to permit access despite their causing certificate errors, place those sites in the Trusted Sites zone in IE and this will permit access to the sites despite the errors.

Certificate Revocation Checking

In Internet Explorer 8.0 and later, and with the Edge browser, certificate revocation checking is enabled by default, both for the web server's certificate and the issuing Certification Authority's (CA) certificate. It should be left enabled.

Trusted Certification Authorities

Which root Certification Authorities (CAs) should our computers trust? This was discussed in the PKI manual this week, but, most importantly, we need to be prepared to remove a root CA certificate quickly if it turns out that the CA has been compromised. Microsoft might release a patch for this or we may need to push out a script through Group Policy to remove the root CA certificate ourselves.



Kerberos Overview

The default authentication protocol for domain-joined Windows devices is Kerberos. This means that, whenever possible, Kerberos authentication will be negotiated between client and server instead of NTLM, but NTLM is still available for backwards compatibility. Virtually all communications between domain controllers and member computers is authenticated with Kerberos.

Kerberos Benefits

Kerberos enjoys the following benefits over NTLM:

- Kerberos is faster than NTLM and consumes less bandwidth. NTLM is an RPC-based protocol, while Kerberos uses UDP (for the most part). NTLM requires pass-through of authentication traffic to a domain controller; Kerberos clients, on the other hand, can reuse their locally-cached tickets for hours and can send them to servers without the necessity of those servers contacting any domain controllers.

- Kerberos uses mutual authentication, NTLM only authenticates the client.

- Kerberos can be used with Smart Cards for two-factor authentication (PKINIT) where the TGT is encrypted with the user's public key.

- Kerberos credentials can be forwarded or "delegated", while NTLM supports only standard one-hop impersonation. This assists distributed applications enforce user-based access control and supports auditing of that user at each server.

- Kerberos is an RFC 1510 standard, hence, it offers the potential for better interoperability with non-Microsoft hosts, such as Linux.

Brute-Force Attacks Against The Password

However, Kerberos authentication traffic can be "sniffed" and then a brute-force attack mounted to attempt to reveal the user's password. This is possible because, ultimately, the user's master key is derived from the user's password, and this key is used to encrypt sniffable data during the AS Exchange (see below). As always, strong password policy is required.

The original Kerberos sniffer-and-cracker for Windows is KerbCrack (ntsecurity.nu), but this capability is built into several other tools now.

Kerberos Tools

There are a variety of tools and Group Policy settings related to Kerberos. Most have to be separately downloaded from Microsoft, where most Resource Kit and Support Tools have migrated today. If a tool seems to be missing, please check that you have the necessary software installed.

KERBTRAY.EXE (*Resource Kit*) -- Shows a taskbar icon for displaying and purging Kerberos tickets. Ticket flags, lifetimes, and other details are shown in a user-friendly format.

KLIST.EXE (*Resource Kit*) -- Command-line tool for displaying and purging tickets one-by-one. Shows the raw hex ticket flag fields.

NETDOM.EXE (Support Tool) -- Command-line tool which can 1) join a computer to a domain, 2) manage computer accounts, 3) establish trust relationships, including trust links to non-Microsoft Kerberos realms, 4) verify and test NetLogon secure channels, 5) manage the transitivity characteristic of a trust relationship.

SETSPN.EXE (*Resource Kit*) -- Manage the Service Principal Names (SPNs) of Kerberized services running on hosts with computer accounts in AD.

Kerberos RFCs and IETF Drafts

The following is a list of RFCs and IETF drafts which Microsoft has proposed, extended, conforms to or to which Microsoft attempts to conform. They are listed here for reference. Search for drafts and RFCs at http://search.ietf.org.

- RFC 1510: Kerberos
- RFC 1964: GSSAPI for Kerberos
- RFC 2478: SPNEGO
- RFC 3244: Windows Kerberos Change Password and Set Password

Kerberos Glossary of Terms

You will often see or hear Kerberos-related terms in security articles. Here is a brief glossary of Kerberos terms for reference.

Key Distribution Center (KDC)
The Key Distribution Center (KDC) is that trusted server or service which issues tickets and manages keys on behalf of other principals. The KDC is made up of two services: the Authentication Service (AS) and the Ticket Granting Service (TGS).

Every AD domain controller is a KDC. Keys are stored in AD and multi-master replicated to all domain controllers, hence, there are no "master" or "slave" KDCs. The AS and TGS services are housed together in the Local Security Authority (LSASS.EXE) running on each domain controller.

The KDC listens on UDP and TCP port 88 for ticket requests. UDP is used by default, but TCP is employed when a ticket is larger than 2KB and can be configured, through a registry modification, to always be used instead of UDP (KB244474).

Authentication Service (AS)
The Authentication Service (AS) is that part of the KDC which issues Ticket Granting Tickets (TGTs) to access the TGS on the KDC. You authenticate once to the AS for a TGT with your secret master key at the beginning, then your master key is not used afterwards until you need a new TGT; this prevents adversaries from sniffing too much data encrypted with your master key, which would happen if you used your master key for every ticket request, and optimizes the performance of the KDC, since it will not have to access your master key for every ticket request you would make afterwards.

Ticket Granting Service (TGS)
The Ticket Granting Service (TGS) is that part of the KDC which issues session tickets. (It also issues TGTs to TGSs in other domains for cross-domain authentication referrals, but that will be discussed later.) To request a ticket from the TGS to another server, a client must present a TGT to the TGS.

Ticket Granting Tickets (TGT)
A "session ticket" is what a client presents to a target principal when authenticating to that principal. A "principal" is just any computer, person, service or *thing* that can engage in Kerberos authentication exchanges.

A ticket has a number of fields, but, most importantly, it contains 1) an encrypted "session key" and 2) an "authenticator". These are discussed below.

A Ticket Granting Ticket (TGT) is just a session ticket to the TGS itself. Clients need a TGT in order to request more tickets from the TGS, and more tickets are needed if the client is to authenticate to other servers on the network. The TGS is like any other service in that you must present a ticket to it first in order to access resources, the only

difference being that a TGS dispenses Kerberos tickets instead of web pages or e-mail messages like "regular" servers. What ticket does the TGS expect? The TGT!

Master Key
The "master key" is a symmetric key unique to a particular principal. A user's master key is derived from that user's password stored in AD, hence, only that user and the domain controllers have knowledge of the user's master key. Master keys are never exchanged with other servers besides KDCs. The master keys of computers are derived from their password hashes, and computers change their passwords automatically every 30 days by default. There may be multiple master keys per principal. The RC4 master key is an MD4 hash of the full case-sensitive Unicode password. The AES master key is derived in part from the user's password hash, but also a salt.

Session Keys and Tickets
A "ticket" is something a client sends to a server in order to authenticate to that server. It includes everything the server needs to identify the client and all the groups the client belongs to so that server can authorize and audit requests made by the client. This information is stored in the "SID PAC" of the ticket. A ticket also includes an encryption key shared between the client and the server, but that session key is not included in the ticket in cleartext. The session key in the ticket in encrypted with the server's master key, so only the server can decrypt the session key in the ticket.

Hence, a "session key" is a symmetric key shared between two principals for the sake of mutual authentication. The session key is randomly generated by the KDC when the client requests a ticket. Two copies of the key are included with the ticket the KDC sends to the client: one copy is encrypted so only the client can read it, the second copy is encrypted so only the target server can read it. Upon receipt from the KDC, the client decrypts its own copy of the session key, creates an "authenticator" with it, then sends the authenticator and the ticket (which includes the server's encrypted copy of the key) to the server. At no time is the session key transmitted over the network in the clear.

The KDC sends the ticket to the client instead of the target server in order to optimize traffic flows and, more importantly, permit the client to cache the ticket. By default, a client can cache and reuse a ticket for 10 hours.

Windows Kerberos session keys can be 56-bit DES, 56-bit RC4, 128-bit RC4, 128-bit AES, or 256-bit AES.

The SIDs PAC
The Privilege Attribute Certificate (PAC) is a field in a ticket which contains all the user's Security ID numbers (SIDs) from the global and universal security groups of which the user is a member. When a ticket is presented to a server, that ticket contains the user's SIDs PAC, hence, that server can authorize and audit that user's requests. More formally, with the PAC data a server can construct a Security Access Token (SAT) to represent the remote user (or computer, if that's who the principal is) and use that token to impersonate the user locally. A SAT includes all the SIDs for a principal --user, global

groups, universal groups, and local groups-- as well as a list of all the "privileges" the user has on the server's machine, e.g., take ownership, debug programs, etc.. With the SAT, the server can authorize access to files and services based on access control lists.

The PAC data is digitally signed by the KDC to ensure its integrity and authenticity.

Authenticator
An "authenticator" is something a client sends to a target principal to prove that the client really is who/what the client proclaims to be. The client will have received a ticket to the principal from the TGS on the KDC, so the client needs to prove to the principal that the client is indeed in possession of a copy of the session key sent to the principal in the ticket.

How can you prove to another party that you have a copy of a symmetric key which is (supposedly) shared only between the two of you, but without sending the key itself? The answer: encrypt something with that key and send the ciphertext to the other party. This ciphertext is the authenticator.

The "authenticator" is some well-formatted data encrypted with a session key shared with another principal; the data is formatted to include the identity of the client, the KDC's identity, and a timestamp accurate to within a fraction of a millisecond of the client's clock. The identities in the authenticator must match the ticket, and the timestamp cannot be too old (no more than five minutes old, by default).

Additionally, all Kerberos principals keep track of the tickets/authenticators they've received recently to guard against replay attacks. Hence, within a, say, five-minute period, if an adversary replays captured tickets and authenticators to a server, that server will notice that the authenticator is a duplicate. The server doesn't have to keep track of too many authenticators because they quickly time-expire.

Principals and Principal Names (UPN and SPN)
A "principal" is a user, computer or service which engages in Kerberos authentication.

A User Principal Name (UPN) identifies a user. It looks like an e-mail address, but the domain in the address is the user's AD domain or realm, e.g., in "amy@sans.org" the username "amy" is an account in AD and "sans.org" is typically the user's AD domain name (but may be a Unix Kerberos realm instead).

You can actually log onto your desktop with your UPN when you hit Ctrl-Alt-Del.

Administrators can change the default UPN domain name to anything they desire in the Properties of the "AD Domains and Trusts" snap-in. Hence, my AD domain might be "sans.org", but an administrator could change my user principal name to "jason@tanzarian.com" and it will still work fine because the DC will simply do an LDAP query to find your account in AD.

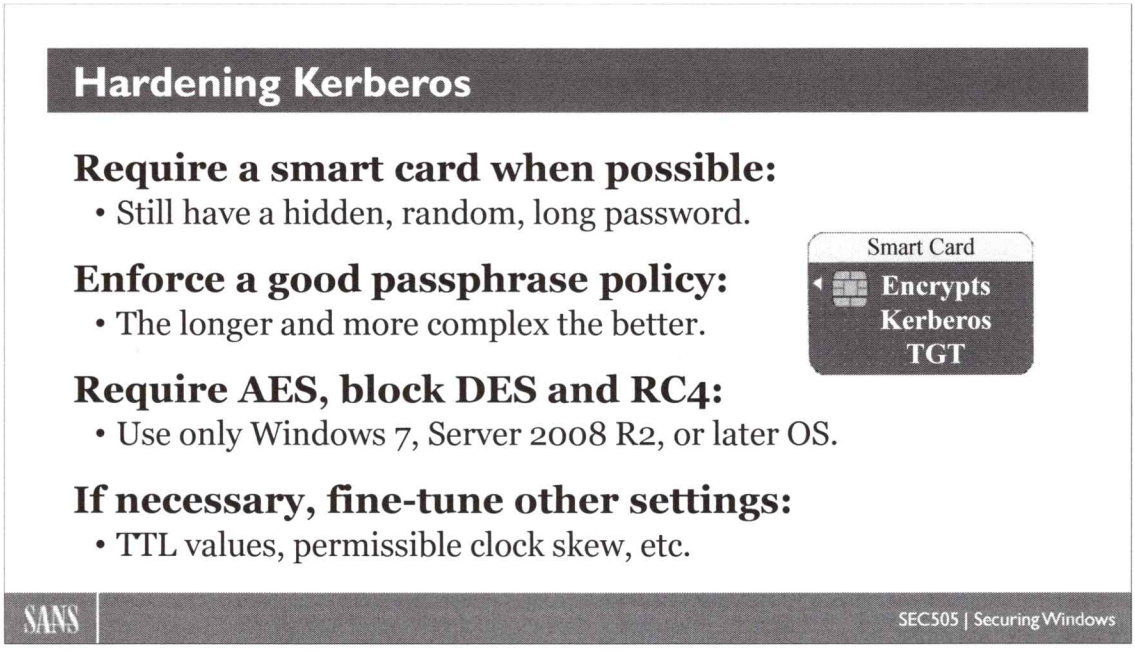
A Service Principal Name (SPN) identifies a service on a host and optionally its port number; for example, "MSSQLSERVER/machine1.sans.org:1433" is the SPN for Microsoft SQL Server listening on TCP 1433 on machine1.sans.org. Additional names can be added manually with the SETSPN.EXE tool. The generic computer SPN for machine1.sans.org would come in two forms: "HOST/machine1.sans.org" and "MACHINE1$@SANS.ORG".

Assuming that machine1.sans.org is a domain controller too, its KDC SPN would be "krbtgt/machine1.sans.org@SANS.ORG" or just plain "krbtgt/SANS.ORG". In fact, there is a global user account on domain controllers named "krbtgt" for the identity of the KDC, and a derivative of its password is used as the master key for the AS for issuing TGTs. The password is automatically updated periodically. The krbtgt account cannot be renamed or deleted.

Realm

On a Windows network, the "Kerberos realm" is identical to your "Windows domain", and the name of your realm is identical to the DNS name of your domain. You will often see the word "realm" used interchangeably with "domain" when discussing AD and Kerberos.

Note that Windows currently supports only Kerberos version 5, not version 4.

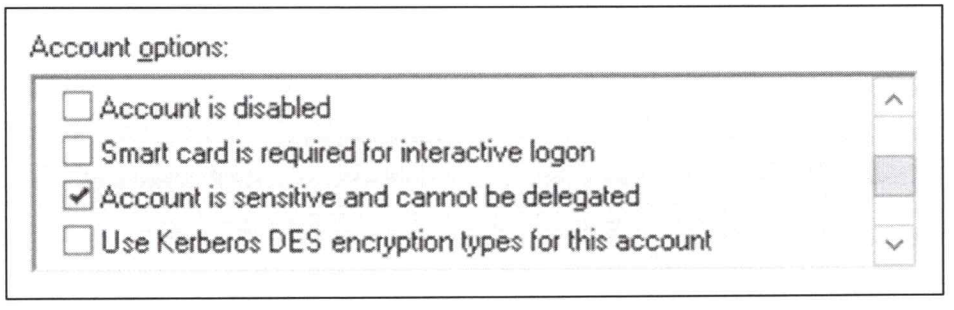


Hardening Kerberos

Kerberos is a great authentication protocol, but not perfect. Kerberos is vulnerable to various cracking and MITM attacks. Remember, Kerberos is only as strong as the length and randomness of the user's or computer's passphrase. Computer passwords are randomly generated, over 100 characters long, and reset automatically every 30 days. We need to mostly worry about Kerberos for user accounts. And not just human users, there are also user accounts for services and scheduled jobs too.

Require Smart Card Authentication When Possible

Recall from a different manual in this course that, when a user's account is configured to require a smart card for interactive logon, that user's password is reset to a 100+ character random password, hence, the hash of that password is virtually impossible to crack. For high-value targets like administrators, use your PKI to issue them smart cards and require them for interactive logons.



For added security, a scheduled script can be used to automatically uncheck the box to require a smart card for interactive logons of these smart-card-required accounts, assign a

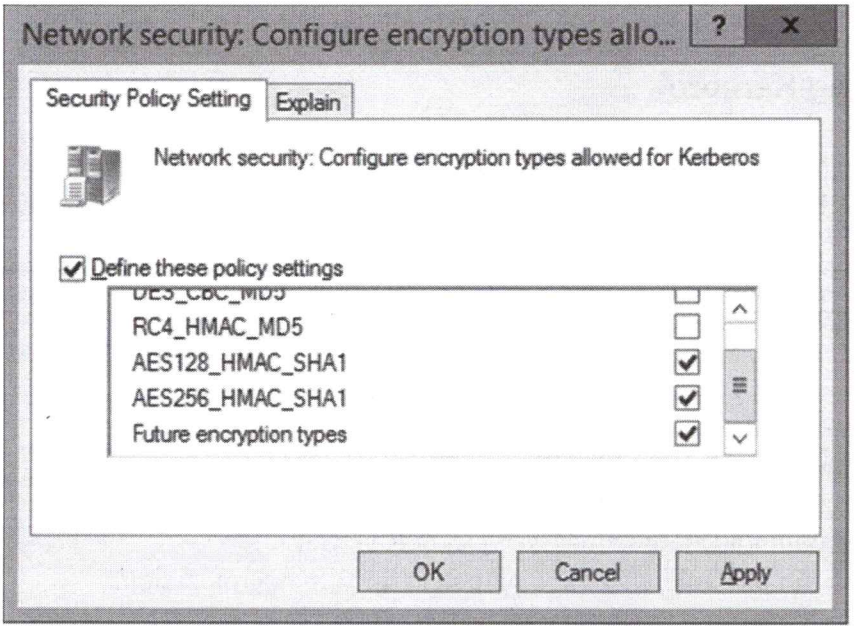
100-character random password, then re-check that box again (which resets again). Have the script run when it's least likely for these admins to be logged in anywhere.

Enforce Passphrase Policy

Deploying smart cards is not always possible. At a minimum, require a good passphrase policy for all users, especially for high-value targets. Different groups may have different password policies assigned to them: the more powerful the group, the longer the passphrase must be. As discussed in a different manual this week, aim for a minimum of 15 characters for high-value users.

Require AES, Block DES And RC4

DES and RC4 are obsolete ciphers. On Windows 7, Server 2008 R2 and later operating systems, it's possible to set a machine-wide policy to block DES and RC4 for Kerberos. It is also possible to require a 256-bit key for AES instead of just 128 bits.



In a GPO, go to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options. There you will find a setting named "Network security: Configure encryption types allowed for Kerberos" with a list of cipher suites. Here are the checkbox options:

Cipher Suite	Checked or Unchecked
DES-CBC-CRC	Never Checked
DES-CBC-MD5	Never Checked
RC4-HMAC	Possibly Checked
AES128-CTS-HMAC-SHA1-96	Possibly Checked
AES256-CTS-HMAC-SHA1-96	Checked
Future encryption types	Checked

DES should never be used. You may be compelled to use RC4 if you still have Windows XP or Server 2003, but these operating systems are ancient, so it's best to upgrade them. Windows Vista and Server 2008 later support AES by default, even if they lack the particular GPO setting we are currently discussing to require AES; nonetheless, they do support AES out of the box. Finally, 128-bit AES may be fine for several years, but given the advancements in quantum computing and how cheap it is to rent megaclusters of cracking nodes from cloud providers, you might as well require 256-bit AES. It will be very rare for a Windows or non-Windows machine to support 128-bit AES but not also 256-bit AES.

(You may have read about user and computer attributes named UserAccountControl and msDC-SupportedEncryptionTypes for managing Kerberos encryption options. Once the above GPO policy is set on a computer, these attributes in AD are ignored. These attributes are really only for legacy operating systems. Nonetheless, if you do have a large number of Windows Vista or Server 2008 hosts for some reason, you may wish to write a PowerShell script to manage these AD attributes.)

Fine-Tuning Kerberos

The operation of Kerberos can be fine-tuned through Group Policy Objects (GPOs) linked to the domain container. Note that any GPO managing one of these Kerberos policies must be linked to the domain itself, not to the Domain Controllers OU or anywhere else.

These Kerberos GPO options are mostly located under Computer Configuration > Policies > Windows Settings > Security Settings > Account Policies > Kerberos Policy:

- Enforce User Logon Restrictions.
- Maximum Lifetime for Service Ticket.
- Maximum Lifetime for User Ticket.
- Maximum Lifetime for User Ticket Renewal.
- Maximum Tolerance for Computer Clock Synchronization.

Enforce User Logon Restrictions

If "Enforce User Logon Restrictions" is enabled, the KDC will perform a few checks before issuing a ticket to a user. The KDC will check that the user's account has not been deleted or disabled, and that the user has the necessary logon right for the type of access being requested, e.g., over the network or interactive logon.

The security benefits are clear, but how the checks are performed and on what operating system versions is complex (see the official MS-KILE specifications). Because of the performance penalty, some checks are only performed on TGTs older than 20 minutes, and other TGTs, such as for services running as Local System, are not checked at all. Nonetheless, the default is enabled and the recommendation is to leave it enabled.

Ticket Lifetimes

Kerberos tickets cannot be cached and reused forever. To determine the appropriate time to live (TTL) of various ticket types, you must balance security with performance, especially when DCs are servicing a large numbers of clients.

- **"Maximum Lifetime for Service Ticket"** specifies how long cached tickets are permitted to be replayed to target servers. When the TTL expires, the client will have to perform another TGS Exchange to get a new ticket and session key to the desired server. Note that the client's current connections to the target server will not be terminated when the ticket expires, rather, any *new* authenticated connections will require a TGS Exchange after the relevant cached ticket expires. The shorter the lifetime, the better the security, but also the worse the performance of the network as more TGS Exchanges are now required. This must be ten minutes or greater, but not longer than the maximum user ticket lifetime. The default is 600 minutes (10 hours). Recommendation: 60 minutes in medium- to high-security environments, 600 minutes in low-security environments.

- **"Maximum Lifetime for User Ticket"** specifies the TTL of the client's TGT. At expiration, a new TGS Exchange will be performed to renew the TGT's TTL (not an AS Exchange), but the user will not be prompted for a password. The default is 10 hours. Recommended: 1 hour in medium- to high-security environments, 10 hours in low-security environments.

- **"Maximum Lifetime for User Ticket Renewal"** specifies the absolute maximum TTL of a session ticket *or* TGT (even though it says "user ticket", it applies to all ticket types, see KB232179). After expiration, the ticket cannot be renewed and a new ticket must be obtained. The default is 7 days. Recommendation: 7 days.

- **"Maximum Tolerance for Computer Clock Synchronization"** determines the maximum skew between the KDC's and client's clocks after which the KDC will refuse to issue tickets. The threat is that an adversary may attempt to capture and replay tickets at a later time. The default is 5 minutes. Recommendation: 5 minutes. Be very careful about setting this less than the default.

Preauthentication

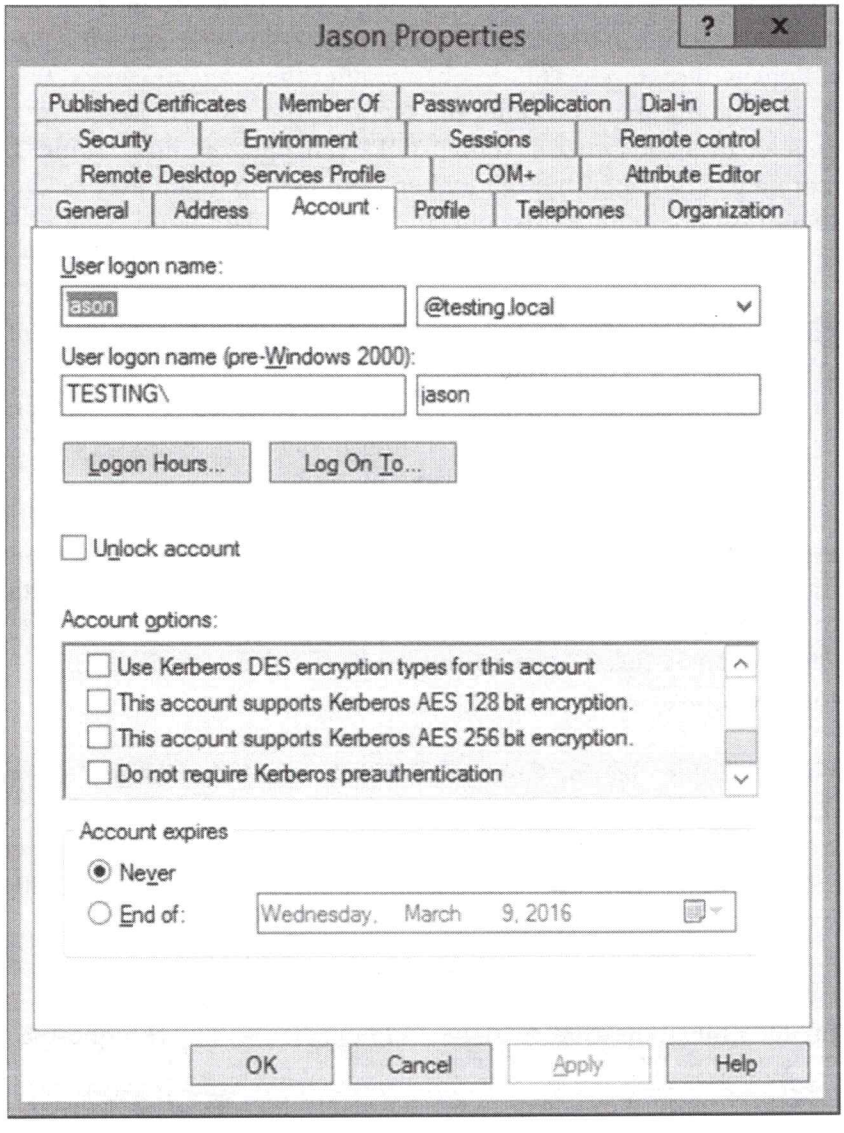
By default, "preauthentication data" is included in the client's initial request to the KDC during the AS Exchange. The purpose of this data is to prove to the KDC that the client knows the password (master key) prior to the AS returning a TGT. This is an important security requirement because, without it, an adversary could request millions of TGTs for a user and attempt a brute-force key search against them all (the more TGTs retrieved, the better the odds the brute-force attack will succeed). Also, simply requesting millions of TGTs is a denial of service attack in itself.

Preauthentication is required by default and can only be disabled on a per-user basis. Preauthentication cannot be disabled for an entire domain or OU through Group Policy.

If large numbers of users need this change, write a custom PowerShell script. Disabling the preauthentication requirement as a permanent fix, however, is not recommended.

Try It Now!

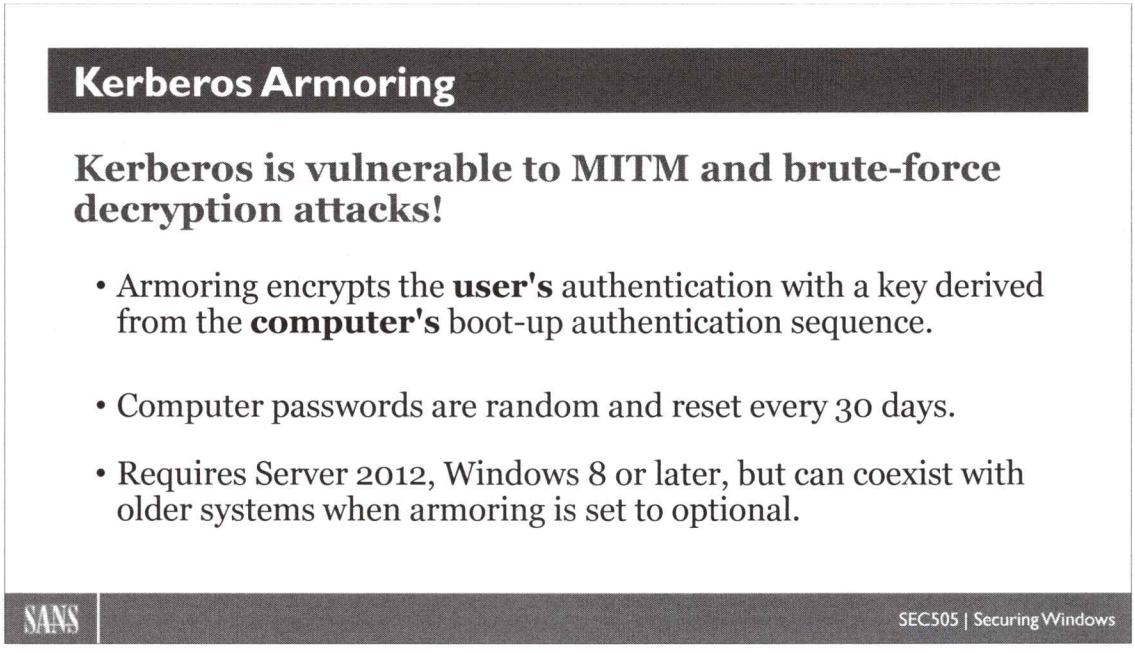
To disable the requirement for preauthentication data on a particular user account, right-click that account in "AD Users and Computers" > Properties > Account tab > check box labeled "Do not require Kerberos preauthentication" > OK.



In the screenshot above, there are three options that should be ignored:

- Use Kerberos DES encryption types for this account
- This account supports Kerberos AES 128 bit encryption
- This account supports Kerberos AES 256 bit encryption

These options control the values assigned to two attributes named UserAccountControl and msDC-SupportedEncryptionTypes. Ignore these attributes and their checkboxes in the screenshot and instead use the machine-wide GPO setting discussed earlier. These attributes are really only for legacy operating systems. Nonetheless, if you do have a large number of Windows Vista or Server 2008 hosts for some reason, you may wish to write a PowerShell script to manage these AD attributes. If you have older machines, they should be upgraded.



Kerberos Armoring

Windows 8, Server 2012 and later operating systems support Flexible Authentication Secure Tunneling (FAST) for Kerberos, also known as "Kerberos armoring", as defined in RFC 6113. Kerberos is vulnerable to packet sniffing and offline dictionary attacks. Kerberos error messages can also be spoofed to downgrade to weaker keys or even to downgrade to NTLM. FAST is designed to combat these offline attacks and these downgrade vulnerabilities. FAST itself does not authenticate the client to the domain controller, instead, it provides a means to encrypt the client's pre-authentication data to the controller to make offline attacks more difficult. Depending on implementation details, FAST can also help thwart active man-in-the-middle attacks against Kerberos too.

How does it work? When a domain-joined computer boots up, the computer (not the user yet) will authenticate to a controller using standard unarmored Kerberos and obtain a session key using its Ticket Granting Ticket (TGT). With session key in hand, the computer will encrypt the user's pre-authentication data when a user later logs onto the computer. In short, the user's authentication sequence is protected with a key first derived during the computer's logon, but the computer's logon sequence is itself not secured using this technique. Because users' passwords are typically much easier to crack than the randomly-generated passwords used by computers, we still get a net increase in security.

> **Note:** There is another Kerberos armoring method which is supported too, namely "anonymous Kerberos" with PKINIT (RFC 6112), but this is not preferred because it is more vulnerable to man-in-the-middle attacks (not discussed here).

Requirements

Only Server 2012, Windows 8 and later operating systems support Kerberos armoring. Only domain controllers running Server 2012 or later support armoring, but having a mixture of older operating systems on the controllers is permitted.

Kerberos armoring is supported across domain and forest boundaries too. However, in a multi-domain or multi-forest scenario, all the controllers in the root domain of the forest(s) must be running Server 2012 or later. At least two controllers in each of the subdomains, if any, should be running Server 2012 or later as well.

(Note that to use claims-based Dynamic Access Control (DAC), Kerberos armoring must be used on the clients and servers where DAC is desired. DAC without the use of user or device claims does not require armoring.)

Older and incompatible operating systems may coexist with armoring-capable systems because Server 2012 and later domain controllers can be configured to merely support armoring as an optional feature instead of making armoring mandatory. Once all systems have been upgraded, Kerberos armoring should be made mandatory.

How To Enable Kerberos Armoring On Domain Controllers

Kerberos armoring is enabled through Group Policy. First apply the change to the domain controllers, then to the rest of the domain (or to the desired OUs).

> ***Try It Now!***
>
> To enable Kerberos armoring on domain controllers, open the "Default Domain Controllers Policy" GPO linked to the domain controllers OU > Computer Configuration > Policies > Administrative Templates > System > KDC, then set "KDC support for claims, compound authentication and Kerberos armoring" to Enabled and then select Supported. You must reboot.

There are four options when configuring Kerberos armoring on domain controllers:

- **Not supported**: Kerberos armoring is disabled. This is the default.

- **Supported**: Kerberos armoring is optional, the client chooses whether to request it. This option is backwards compatible with pre-Windows 8 clients. When the domain functional level is Server 2008-R2 or lower, this GPO setting can only be set to Not Supported or Supported (even if one of the following options is chosen in the GPO graphical interface, the following options are only interpreted as Supported until after the domain functional level is raised to at least Server 2012).

- **Always provide claims**: Kerberos armoring is optional, but claims information will always be returned for the sake of Dynamic Access Control. Requires the domain functional level to be at least Server 2012, hence, all controllers in the domain must be running at least Server 2012.

- **Fail unarmored authentication requests**: Kerberos armoring is mandatory. Requires clients running Windows 8, Server 2012, or later. Requires the domain functional level to be at least Server 2012, hence, all controllers in the domain must be running at least Server 2012.

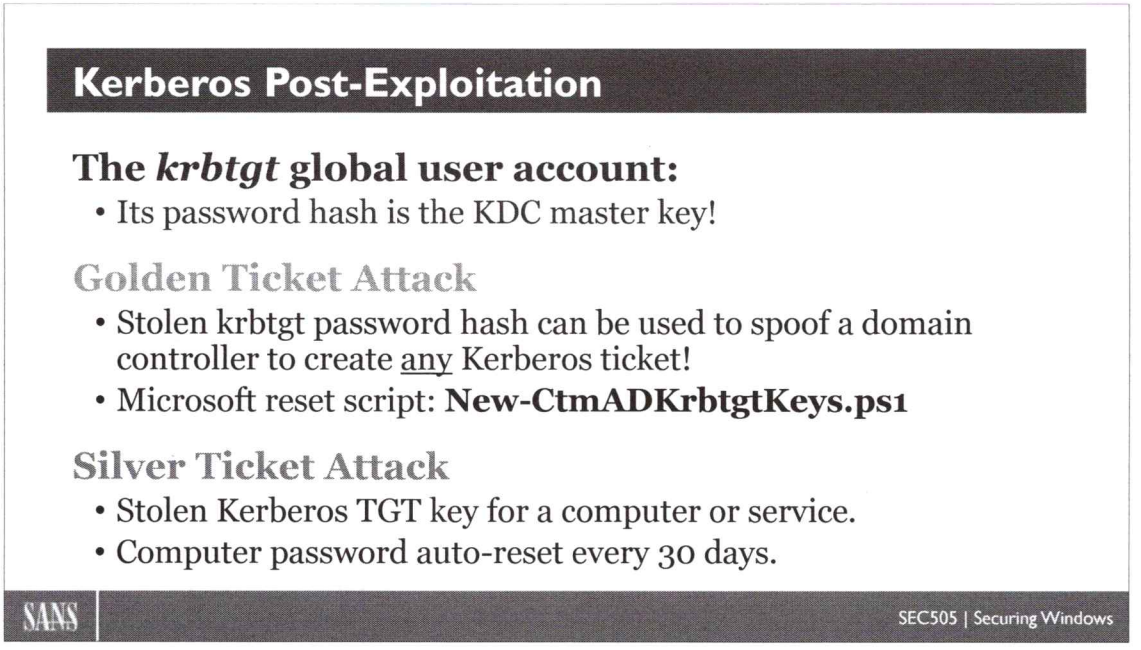
Domain controllers earlier than Server 2012 ignore these Group Policy settings for Kerberos armoring. You do not have to upgrade all controllers simultaneously to Server 2012 or better.

How To Enable On Client Computers (On Non-Domain Controllers)

You must also enable Kerberos armoring on domain-joined client computers, which is a separate step from enabling armoring on the controllers.

> ***Try It Now!***
>
> To enable Kerberos armoring on all client computers in the domain, edit the "Default Domain Policy" GPO linked to the domain > Computer Configuration > Policies > Administrative Templates > System > Kerberos > then set "Kerberos client support for claims, compound authentication and Kerberos armoring" to Enabled. (This does not have to be enabled for the entire domain, but usually it will be.) Reboot required.

Once armoring is enabled, there is nothing different to see in the graphical interface of a client computer. No user training is required.



Kerberos Post-Exploitation

An Active Directory domain controller acts as a Kerberos Key Distribution Center (KDC). As a KDC, it creates Kerberos authentication tickets for use in the domain/realm. As a KDC, a controller must be in possession of a special encryption key used to protect and sign ticket data. There is not just one copy of this key. Every domain controller is a KDC, so every domain controller needs a copy.

So, where is the key? The key is the password hash of a user account named "krbtgt." The krbtgt account is created automatically when the first controller in a domain is installed. Except when a domain's functionality level is raised from the 2003 level to something higher, which is a one-time event, the password of the krbtgt account is never changed. There is no automatic update or reset of this password!

The krbtgt password hash is an MD4 hash of the UTF16LE-encoded password of the krbtgt account (the so-called "NT hash" of the Unicode password). This is used with RC4 ticket encryption. But there is actually three other keys used by krbtgt for Kerberos. These other keys are 56-bit DES, 128-bit AES, and 256-bit AES. The AES keys are salted at least, but the others are not salted.

The krbtgt account keys are used to 1) encrypt the Ticket Granting Ticket (TGT) given to a user or computer after initial authentication and 2) to digitally sign the Privilege Attribute Certificate (PAC) inside that TGT. The PAC contains identifying information about that user or computer, such as their global group memberships in Active Directory.

Domain controllers also cache the prior set of krbtgt keys, just in case the current krbtgt keys are changed. Hence, there is always two sets: the current set and the prior set. As we will see in a moment, this is important.

Golden Tickets

If an adversary can steal the password hash of the krbtgt account, that adversary can act as a KDC for the domain too. This means the adversary could create Kerberos authentication tickets with any data whatsoever; for example, a hacker could create an authentication ticket for him- or herself with membership in the Domain Admins group. This spoofed ticket is called a "Golden Ticket" because it can be used to access any resource anywhere in the domain!

Keys for krbtgt can be stolen from the live memory of a domain controller, from the NTDS.DIT file from a controller (this is the AD database itself), or by spoofing an apparently-legitimate controller on the LAN and tricking a real controller into replicating the krbtgt keys to the spoofed control via normal AD replication. A copy of the NTDS.DIT file could be obtained from the drive of a running controller, a backup, an exported VM, or a VM being copied or synced over the network. In general, there are several techniques, both existing today and theoretically exploitable in the future, for obtaining copies of the krbtgt keys.

Once in possession of the krbtgt keys, an attacker could use a free tool like Mimikatz to spoof Kerberos tickets for any purpose whatsoever (https://github.com/gentilkiwi).

Microsoft KRBTGT Password Reset Script

Like any other service account, the password of the krbtgt account should be changed on a regular basis, perhaps monthly or quarterly. The krbtgt password must also be reset when there is evidence of a domain controller compromise.

Resetting the krbtgt password must be done carefully to avoid disruption. Microsoft has a PowerShell script to reset the krbtgt password in an emergency or for a regular monthly hygiene reset. An emergency reset is far more likely to cause problems.

To find the latest download URL for the script, do a search on "KRBTGT Account Password Reset Script" or the name of the script itself, which is "New-CtmADKrbtgtKeys.ps1". Make sure to read the usage notes for the tool, such as the audit mode and potential problems, especially with emergency resets.

The script requires RPC access to all domain controllers. Run the script on the PDC emulator as a Domain Admin member. The script can check your readiness to perform the reset in an audit-only mode (this is "Mode 1"). Note that the script is only supported at the 2008 domain functional level or higher. The script writes to its own text log file.

The script will reset the krbtgt password and trigger an immediate AD replication. To avoid most potential problems, you would then wait at least 10 hours (the default session ticket lifetime) and then run the script again. In an emergency, the script would be run

twice in a row without delay. Resetting the krbtgt password twice is required because controllers will cache (and use) the prior password/keys.

The emergency no-delay double reset will be more disruptive, possibly requiring user logoff/logon and a reboot of servers, especially if those servers have services running under custom accounts or if they have older services from third-party vendors. The wait-10-hours safer method allows a longer window of re-attack, but will have much lesser impact, possibly with no disruption whatsoever, depending on services design.

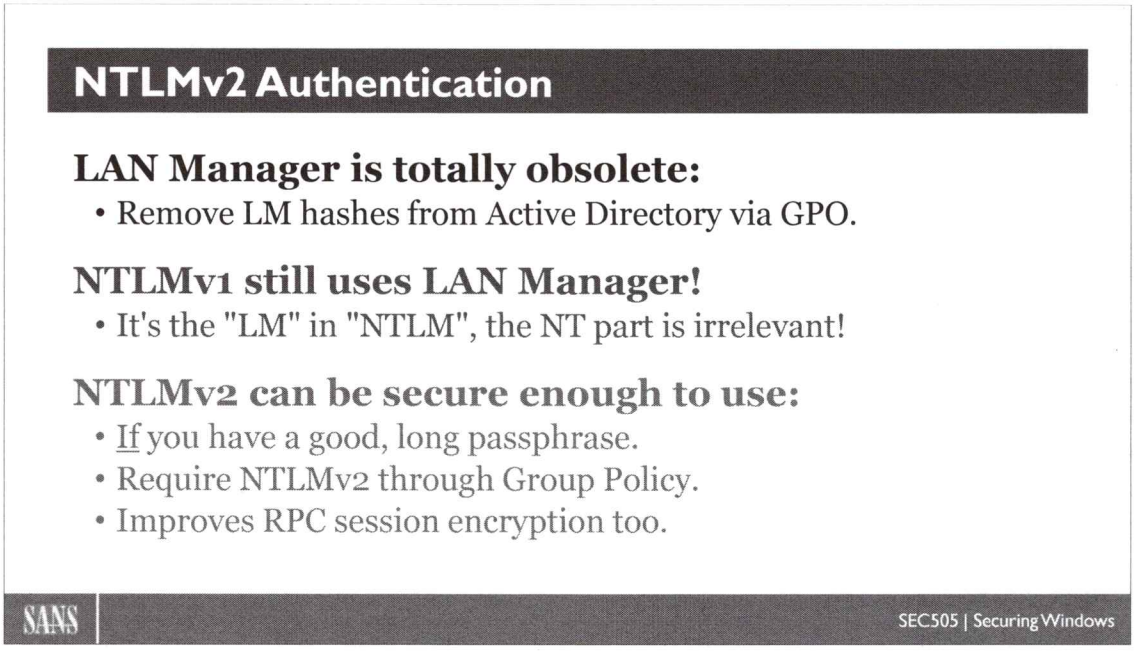
After a compromise, keep in mind that a krbtgt password reset would be only one part of a larger domain recover effort. All user and computer account passwords, in this case, will need to be reset (and computer account passwords twice). Incident responders will need to look for other malware, block command-and-control packets, reset other encryption keys, restore from backups, etc. A domain controller compromise is almost a worst case scenario, so recovery will require a lot of work.

Silver Tickets

With a stolen krbtgt key, an attacker can create a Golden Ticket. With the password hash of a non-domain-controller computer account or service account, an attacker can create a Silver Ticket instead. Silver Tickets can only be used to authenticate to the computer or service whose password hash or Kerberos TGT key has been compromised. For example, if an old server was still using 56-bit DES encryption for its Ticket-Granting Ticket from the controller/KDC, then the encryption could be brute-forced, allowing the attacker to authenticate to that server with any identity or group memberships desired, such as Domain Admins. Only that one server is affected, so it's must less worse than a universal Golden Ticket, but it's still bad enough, especially if it's a mission-critical server.

Fortunately, computer account passwords/keys are reset automatically every 30 days. Except in an emergency, there's nothing we have to do. In an emergency, the compromised server would likely need to be disjoined from the domain, disconnected from the network, rebuilt, connected back to the domain, and it's computer account password immediately reset with the Reset-ComputerMachinePassword cmdlet or the old NETDOM.EXE tool.

However, the passwords for service accounts, on the other hand, are not reset automatically, except with Managed Service Accounts (MSAs). As discussed elsewhere in this course, non-MSA service account passwords should also be changed periodically too, perhaps quarterly, and they should certainly be reset after a compromise.



NTLMv2 Authentication

Even though Kerberos is the default and preferred authentication protocol, the "Windows NT/LanManager" (NTLM) and "LanManager" (LM) protocols are still supported for backwards compatibility and for when Kerberos cannot be used, e.g., Windows Telnet does not support Kerberos, but it can use NTLM.

When password information is transmitted across the network with NTLM, the password itself is not sent. Rather, a hash of the password is used as an encryption key for encrypting a random challenge sent from the server to the client. When the server encrypts the challenge with the user's password hash, and the result on the server is identical to what the client returned, this proves that the user knows the password.

> **Tip:** You can track NTLM authentications with Performance Monitor on domain controllers (NTDS:NTLM Authentications).

The AD database and the local SAM accounts database both store a user's password information in a hashed format. Two hashes are stored: the LM hash and NTLM hash.

Two Versions of NTLM

NTLM comes in two versions: NTLMv1 and NTLMv2. They have drastically different security characteristics. For example, NTLMv1 traffic can be sniffed by tools like Cain & Abel to extract raw password hashes, NTLMv2 cannot (though other information can be extracted, just not the raw password hashes).

NTLMv2 can also be used to derive shared secret encryption keys between client and server in a way which is much more secure than NTLMv1 (discussed later).

The LM Password Hash Used In NTLMv1 And In Generic LM

The LanManager (LM) hash used in NTLMv1 is derived by the following steps:

1. Make the cleartext password 14 characters long by either truncating it or padding null characters (blanks).
2. Convert all characters to uppercase. Numbers and symbols are unchanged.
3. Reverse the bits of each character byte.
4. Split the 14-byte string into two 7-byte pieces.
5. Use each 7-byte piece as an encryption key with the DES algorithm to encrypt a fixed string built into the operating system. This "magic string" is the same on all Windows computers and is 8 bytes in length (0x4B47532140232425 hex).
6. Take the two results of encrypting the magic string with the two DES keys and concatenate them together. This produces a single 16-byte output. This 16-byte output is the LanManager hash of the password.

Weaknesses of the LanManager hash that makes it easier to crack:

- Passwords longer than 14 characters are no stronger than 14-character passwords because they are truncated.
- Case-sensitivity is lost because the password is converted to uppercase. This reduces the "keyspace" which needs to be searched to break the encryption because there are fewer possible characters.
- Because the 14-byte password is broken into two 7-byte strings, and these strings are used to separately encrypt the magic string, this reduces the effective length of the encryption key from 14 to 7 bytes (a reduction in encryption strength by orders of magnitude).
- If the original password is seven characters or less, this is instantly recognizable because the second DES key will be all null characters (blanks).
- Because no random bytes are added to the hashing process (no "salt"), two identical passwords will have identical LanManager hashes. This allows an attacker to use a pre-computed dictionary of LanManager hashes to quickly search for matches.

When used during NTLMv1 challenge/response authentication, the 16-byte LanManager hash is padded with five bytes of "0" (zero) characters to produce the 21-byte session key. This is cut into three 7-byte pieces which are each used as DES-CBC keys to encrypt the challenge. The three encrypted challenges are concatenated and returned to the DC as the client's response. It is a simple problem to "reverse out" the original LM hash given the server's challenge and the client's response.

The NT Password Hash Used In NTLMv1

The NT hash is produced by truncating or padding the password to 14 characters. This string is converted to Unicode, which preserves the case of the letters. This 14-byte string is hashed with MD4 to produce a 16-byte output. This output is the NT hash of the password.

The NT hash is many times stronger than the LM hash because the full password length is used and case-sensitivity is retained. However, because no "salt" added to the NT hash, two identical passwords will yield identical hashes, hence, pre-computed dictionaries of hashes can be used to quickly crack the encryption. Moreover, because both the NT and LM hashes are used in parallel during NTLMv1 authentication, the strength of the NT hash is *irrelevant* in NTLMv1.

For NTLMv1 challenge/response authentication, the NT hash is padded with 0's to create the 21-byte session key used to DES-encrypt the challenge the same way the LM hash is used (see above). It is also easy to compute the NT hash from the server's challenge and the client's response, though still much more difficult to crack than the LM hash.

NTLMv2 Authentication Details

NTLMv2 also uses a challenge/response method of confirming the user's password, but it is much more secure. A summary of the features of NTLMv2 are:

- The LM hash of the user's password plays no role whatsoever.
- Client input into the challenge (a salt) to prevent chosen plain text attacks. This prevents the use of pre-computed databases of hashes/keys to speed the cracking. The client's response is different with each session even if the user's password stays the same.
- Use of MD5 hashing with salt to prevent "reversing out" the raw password hash.
- Use of a timestamp to verify that the response is timely.
- Client challenge sent to the server to ensure that the server is in possession of the client's password. Strictly speaking, this is not mutual authentication because a rogue server could also have the client's password hash, but it is better than NTLMv1, which made no such check.
- The most efficient cracking method is a dictionary/brute force search of all possible passwords, which is infeasible for long passphrases.
- Man-in-the-middle (MITM) attacks infeasible because of the quasi-mutual authentication. (If an attacker has the password hash already, why do a MITM?)
- Replay attacks infeasible because server's challenge is different each time.
- Downgrade attacks can be prevented by registry values that will require NTLMv2 from either the server's or client's side.
- L0phtCrack cannot extract password hashes from NTLMv2 authentication sessions. Nor is any future version expected to be able to do so.
- It is compatible with 127-character passwords.

The client's response to an NTLMv2 server's challenge is computed as follows:

Client's Response = HMAC-MD5(K, server's challenge + M) + M

Where K = HMAC-MD5(MD4(password), username + domain)
M = ResponseType + HiResponseType + time + client's challenge to server + servername + server's domain.

HMAC-MD5 produces a hash that requires a key (K) to compute the hash. Because the response includes the time and a random challenge to the server, the response is effectively "salted". When computing K, the "MD4(password)" is treated as the HMAC secret key.

> **Note**: Because of the use of a timestamp in the client's response, the date and times on client and domain controller need to be synchronized to within 30 minutes of each other for NTLMv2, and within 5 minutes for Kerberos. Use W32TM.EXE or the NET.EXE TIME command to synchronize clocks.

Configuring NTLMv2 Authentication

Authentication and session security options are set in a registry value named LMCompatibilityLevel, which is located here:

Hive: HKEY_LOCAL_MACHINE
Key: \System\CurrentControlSet\Control\Lsa\
Value Name: LMCompatibilityLevel
Value Type: REG_DWORD
Value Data: 0 to 5

LMCompatibilityLevel can be set to a single digit number between 0 and 5. The following table describes the results of these settings for authentication. (See the second table below for this value's effect upon session security, i.e., on integrity-checking and encryption.) A domain controller is also a client in that one can log on locally and it can be used to access network resources. Systems which are not domain controllers only make use of the client changes specified below.

Level Number	Results of setting LMCompatibilityLevel for authentication.
0	This is the default behavior if the value is not defined. The client will authenticate exactly as it did before and not use NTLMv2. Domain controllers will accept NTLMv2 if a client requests it.
1	Clients will attempt to negotiate NTLMv2, but will fall back to LM and NTLMv1 authentication when necessary. Domain controllers will accept NTLMv2 authentication if the client requests it.
2	Clients will use NTLMv1 authentication only. Clients will not use LM or NTLMv2 responses. Domain controllers will accept NTLMv2 authentication if the client requests it.
3	Clients will use NTLMv2 authentication only. Domain controllers will accept NTLMv2 authentication if the client requests it.
4	Clients will use NTLMv2 authentication only. Domain controllers will refuse LM authentication, and accept NTLMv1 or NTLMv2 authentication if the client requests it.
5	Clients will use NTLMv2 authentication only. Domain controllers will refuse LM and NTLMv1 authentication and accept only NTLMv2.

When in doubt, set LMCompatibilityLevel to one (1) for widest compatibility. This will still use NTLMv2 when it is negotiated, but attackers can trick victims into downgrading.

The LMCompatibilityLevel value also controls how session security options are used, i.e., integrity-checking and encryption of data other than passwords. The following table describes the session security options set by the same LMCompatibilityLevel value above used for authentication.

Level Number	Results of setting LMCompatibilityLevel for session security.
0	NTLMv2 session security is not used. This is the default.
1	Clients use NTLMv2 session security if negotiated with server.
2	Same as 1.
3	Same as 1.
4	Same as 1.
5	Same as 1.

You may also be interested in KB147706: "How to Disable LM Authentication on Windows NT". See KB239869, KB241338 and KB236414 for related information.

Enabling vs. Requiring NTLMv2 Authentication

NTLMv2 support is built into Windows 2000 and later. Nothing needs to be added.

There is also a Group Policy option to *require* NTLMv2 or Kerberos authentication only, and to always reject NTLMv1 and LM. The option is found in the GPO under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options > LAN Manager Authentication Level. In Windows XP and later, the option is named "Network Security: LAN Man Authentication Level" instead.

NTLMv2 Session Security

When an RPC session is established with the PKT_PRIVACY switch to request data encryption, the encryption cipher and keysize are negotiated. When NTLMv2 is the authentication protocol for the RPC channel, the details of the negotiation process can be managed by you. This capability is important because the value of the data being sent through these channels is often very high, e.g., the NetLogon service secure channel.

> **Note:** For more information about RPC security and the registry values below, see the sections on RPC in these books: *Writing Secure Code, 2nd Edition* by Howard and LeBlanc (Microsoft Press) and *Programming Windows Security* by Brown (Addison Wesley). The first covers a wide variety of security topics and has a more practical orientation, while the second discusses low-level internals.

Key Negotiation

The NTLMv2-enabled client generates a 128-bit RC4 key (the "session key"). The session key is re-keyed every 8192 bytes of data encrypted with it. The session key is not based on the user's password, it is generated on the fly. This is the key used for both encryption and HMAC-MD5 integrity-checking. Integrity-checking can be used without encryption, but encryption always includes integrity-checking.

The session key is sent to the server after it is encrypted with another key (the "key exchange key"). This other key is only used for protecting the session key while it is being sent to the server over the network.

> Key exchange key = HMAC-MD5(K, Client's Response)
> where K and "Client's Response" are defined above in
> the section on NTLMv2 authentication.

The key exchange key is different with every session and is 128-bits long, but it is ultimately derived from the user's password, hence, password policy is critical for the protection of encrypted RPC channels as well (when using NTLMv2). Whenever possible, use a 15+ character passphrase instead of just a password.

Requiring NTLMv2 Strong Encryption For RPC Sessions
Certain characteristics of your NTLM-SSP-negotiated session can be made mandatory for the successful negotiation of the session. If these requirements are not met, the session will fail. There is a registry value for the server-side (NtlmMinServerSec) and another for the client's side (NtlmMinClientSec) of the session. Both values are set under the following key (KB239869, KB147706):

> Key: HKLM\System\CurrentControlSet\Control\LSA\MSV1_0
> Value: NtlmMinClientSec, or, NtlmMinServerSec
> Type: REG_DWORD
> Data: Default is zero, but can be any logical-OR combination of:
>
> 0x00000010 = Require message integrity (HMAC signatures).
> 0x00000020 = Require message encryption.
> 0x00080000 = Require NTLMv2 authentication and session key.
> 0x20000000 = Require 128-bit RC4 session key.
>
> If 0x00000010, the connection will fail if integrity-checking is not negotiated.
> If 0x00000020, the connection will fail if encryption is not negotiated.
> If 0x00080000, the connection will fail if NTLMv2 is not negotiated.
> If 0x20000000, the connection will fail if 128-bit encryption is not negotiated.

Importantly, these options can be combined through logical-OR. For example, if NtlmMinServerSec is set to 0x20080030 on the server, then the client must authenticate with NTLMv2, use 128-bit encryption and integrity-check all messages, or else the session will not be successfully established in the first place.

> **Important!** Just because the NTLM SSP enforces these requirements does not mean that every application or service will use the NTLM SSP (in fact, most will not, since Kerberos is the default). And it does not mean that every NTLM-authenticated connection will use its SSP to derive encryption/integrity keys. And it does not mean that every NTLM-authenticated session will use encryption or integrity-checking for its dataflows at all. These details are up to the original

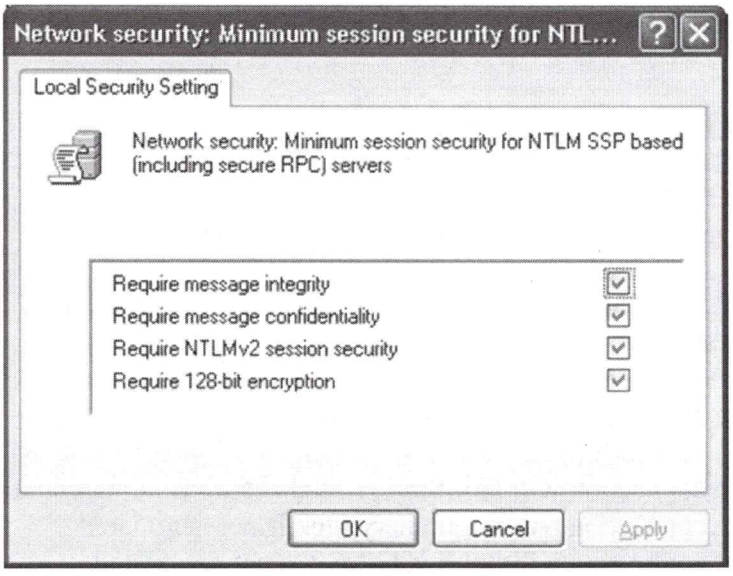
programmer, and if he or she does not provide a mechanism for adding encryption, then nothing can be done about it except to use IPSec.

Also, keep in mind that these settings apply to *all* applications and services that do happen to use NTLMv2 for SSP-derived encryption and integrity keys. Hence, beware of these applications and services breaking when you set security requires too high for their session peers to support, e.g., Windows NT or old versions of Samba on Linux.

Requiring NTLMv2 Strong Encryption For RPC

The registry values above can be set on Windows 2000 and later, but there are also Group Policy options to configure the same, which are easier to manage.

For example, the following screenshot is from a GPO. The setting is found in the GPO under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options > Network Security: Minimum Session Security for NTLM SSP Based (Including RPC) Clients/Servers.



Removing The LM Hash From Active Directory

The LanManager hash can be removed from the Active Directory database entirely with a simple registry change. The LM hash can also be removed from the local SAM accounts database on workstations and member servers.

Removing the LM hash has two benefits: 1) if the AD or SAM database is captured, the LM hashes will not exist to be cracked so easily, and 2) it helps to prevent any use of the LanManager authentication protocol anywhere in the domain.

Consider the following. RainbowCrack is a free password hash cracking tool that can generate a 24GB precomputed database of LanManager hashes for passwords 1 to 7 characters in length from a set of 77 characters. A database of this size cracks 14-character random passwords with 99.9% reliability because of the weaknesses of the LM

hash algorithm. And it does this in *seconds*, not days or months, because database searches are much less computationally expensive than hashing.

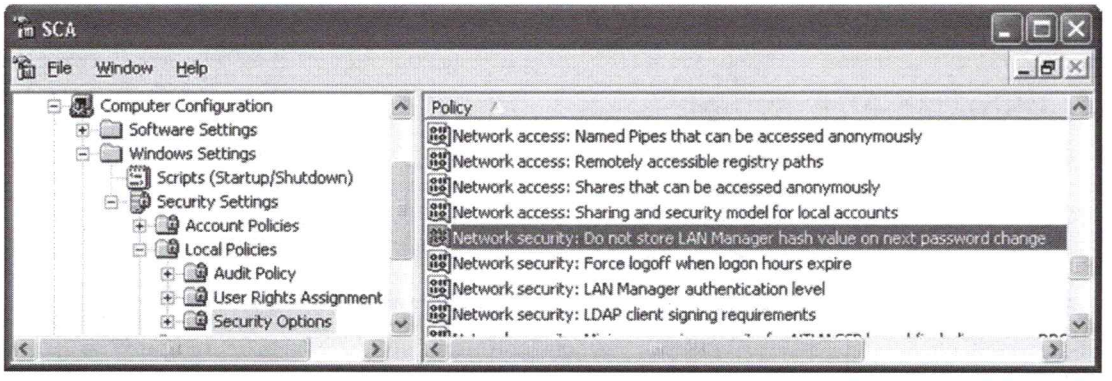
> **Tip:** If your password is 15 characters or longer, Windows doesn't store the correct LM hash of this password anyway. In fact, the LM hash of a 15+ character password indicates that the password is *null*.

Note that he LM hash is not removed from a user's account until the next time the user changes his or her password. Hence, it is not the case that all the LM hashes are simply cleaned out in one sweep.

> **Note**: Install Service Pack 2 or later on Windows 2000 before removing the LM hash from AD or its local SAM database.

How To Remove LM Hash Through Group Policy

There is a Group Policy option named "Network Security: Do not store LAN Manager hash on next password change" which will accomplish exactly what it names. In a GPO, it is located under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options.



Incidentally, if you happen to still have Windows 2000 in production, see KB299656.

Securing The NetLogon "Secure Channel"

The NetLogon service provides a variety of critical domain-related services. Its main RPC channel is called the "NetLogon secure channel" or just "*the* secure channel", but the NetLogon service handles other dataflows and other RPC channels as well. NetLogon runs in the Local Security Authority context (LSASS.EXE).

On member machines, NetLogon assembles the list of all trusted domains for logon, locates domain controllers, handles NTLM authentication of user's domain accounts, and automatically changes the computer's password periodically. One of the first things a member server or workstation does when it boots up is to establish a NetLogon secure channel to a domain controller. When a user changes his or her password, for example, the new password is sent to their domain controller through the NetLogon channel.

On domain controllers, NetLogon handles NTLM pass-through authentication, NTLM trust account management, and SAM replication with Windows NT 4.0 Backup Domain Controllers (BDCs), plus more.

> **Note:** Microsoft has a detailed article on the behavior of the NetLogon service in Windows NT (KB266729). Most of this behavior is retained in later Windows with the important exception that Kerberos is the default authentication protocol.

> **Note:** NETDOM.EXE and NLTEST.EXE can be used to verify and reset the secure channel (KB158148). You can also force the creation of a secure channel to a particular desired controller instead of accepting the auto-selected controller.

Secure Channel Hardening Options

By modifying certain registry values, the secure channel can be integrity-checked and/or encrypted (KB183859). These settings can be managed through Group Policy too. The exact cipher and hashing algorithm used are selected by the underlying Security Support Provider (SSP) which was negotiated to authenticate the RPC connection, hence, the SSP will almost always be NTLM or Kerberos.

All NetLogon channel data can be encrypted and digitally signed for integrity. This is enabled in the registry under the following key:

HKEY_LOCAL_MACHINE
\SYSTEM\CurrentControlSet\Services\Netlogon\Parameters

There are a few values that may be added. Their effects are summarized in the table below. Notice that the signing or encryption is for out-going packets only.

Value Name	Value Type	Effects
SignSecureChannel	REG_DWORD	When set to 1, all outgoing NetLogon channel packets will be digitally signed, but not encrypted.
SealSecureChannel	REG_DWORD	When set to 1, all outgoing NetLogon channel traffic will be encrypted. This option also forces digital signing.
RequireSignOrSeal	REG_DWORD	When set to 1, all outgoing NetLogon channel traffic must at least be digitally signed, but may also be encrypted if this is negotiated. Should a remote system support neither option, the NetLogon connection to it will fail.
RequireStrongKey	REG_DWORD	Require 128-bit or larger encryption key.

These options can be more easily configured through Group Policy. In a GPO, the secure channel options are found under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options:

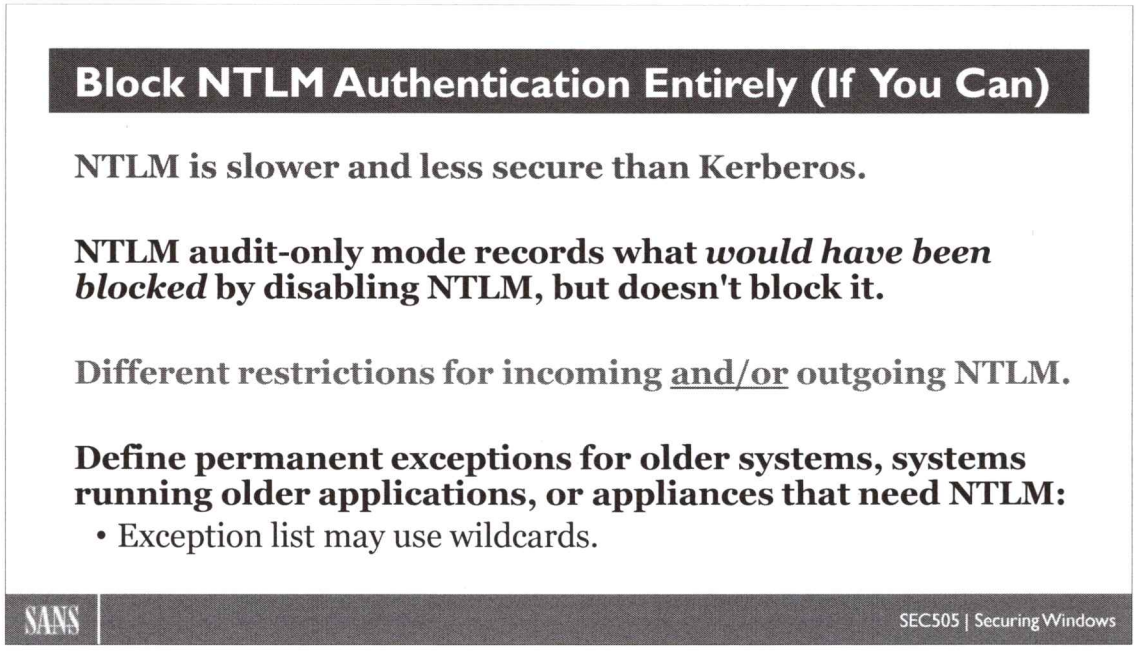
- **Domain Member: Digitally Encrypt or Sign Secure Channel Data (Always).** All domain members must be Windows NT 4.0 SP4 or later.

- **Domain Member: Digitally Encrypt Secure Channel Data (When Possible).** This option is best for backwards compatibility.

- **Domain Member: Digitally Sign Secure Channel Data (When Possible).** This option is best for backwards compatibility and defeating man-in-the-middle attacks, but whenever the secure channel is encrypted, it is also digitally signed.

- **Domain Member: Require Strong (Windows 2000 or Later) Session Key.** This option requires that all domain members must be Windows 2000 or later. The encryption must be at least 128-bit RC4.

The recommendation is to encrypt and sign the secure channel with a strong key, but just be aware of backwards compatibility issues with ancient operating systems and applications.



Block NTLM Authentication Entirely (If You Can)

NTLM authentication is slower, less scalable and less secure than Kerberos. And while NTLM is faster than certificate-based authentication, certificate authentication is much more secure, especially when combined with a smart card. So, just as we are slowly giving up NetBIOS, WINS and LanManager, so in the long run we'll want to migrate away from NTLM in favor of Kerberos and certificate-based authentication too.

Requirements

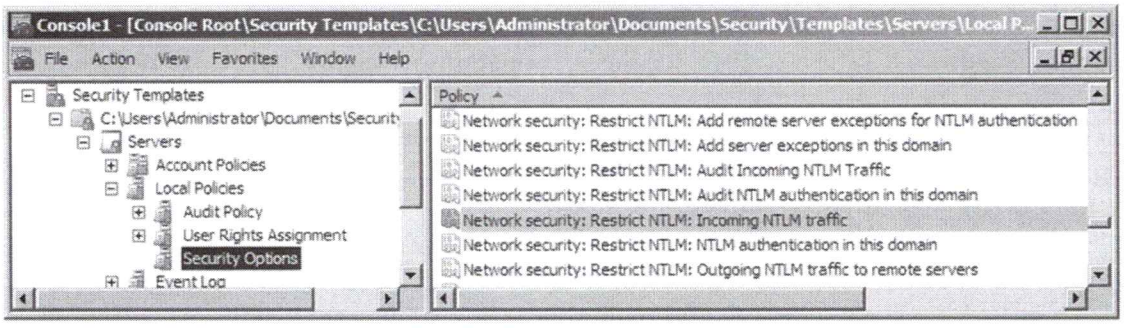
To disable support for NTLM through Group Policy, your systems must be Windows 7, Server 2008-R2 or later, and, to make it practical, all the systems involved should be members of an Active Directory domain.

Configuration

You can find the options to audit and restrict NTLM in a GPO here: Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options. Look for the options that begin with "Network Security: Restrict NTLM".

You can set different policies for inbound versus outbound authentications, as well as merely audit the use of NTLM instead of blocking it.

Notice too that you can define exceptions which allow NTLM by the NetBIOS name and/or fully-qualified domain name (FQDN) of the systems permitted to use NTLM. A wildcard ("*") may be used at the beginning or end of each name in the list of permitted exceptions too, hence, you might implement a naming standard for older servers that require NTLM in order to make defining their exceptions easier.



Deployment Strategy

Restricting NTLM is likely to be used only in locked-down environments where backwards compatibility takes a back seat to security. But eventually NTLM will be disabled by default, so perhaps it's better to begin testing before your hand is forced.

As a deployment strategy, start by using Group Policy to enable NTLM auditing to find out where it is still being used. The aim is to identify which systems are stand-alones, have older operating systems, are running older applications with hard-coded NTLM requirements, or which are not using DNS name resolution but instead connecting by a hard-coded IP address (Kerberos requires knowledge of the name of the other system). Begin now to retire or upgrade systems and applications that need to use NTLM. You should also run protocol analyzers on the network to identify NTLM parties. As you purchase new software, confirm that the software does not require NTLM before signing on the bottom line. This audit, retire, and upgrade process may take two or more years in a medium-sized organization.

When you begin actually restricting NTLM, begin first with some low-value servers and workstations which can be monitored as guinea pigs, then move up to your mission-critical servers and workstations which have been confirmed to not require NTLM in order to gain some immediate security advantage. The myriad of other boxes in between will be configured to block NTLM later.

Inevitably, there will be problems and complaints, so when these occur be prepared to define exceptions in Group Policy if you cannot retire or upgrade the affected machines.

The goal is to not just prefer Kerberos (your domain members already do), but to require Kerberos and to block any tricks other machines might use to downgrade the strength of your authentication to LM or NTLM.

For more information, see: http://blogs.technet.com/askds/archive/2009/10/08/ntlm-blocking-and-you-application-analysis-and-auditing-methodologies-in-windows-7.aspx

Protected Users Group

A special global group named "Protected Users" is created automatically when the PDC Emulator operations master role is hosted on a domain controller running Server 2012-R2 or later. When administrative users are added to the Protected Users group, additional

security restrictions are applied to those members when they log on locally at a computer running Windows 8.1, Server 2012-R2, or later operating systems.

To use the Protected Users group for additional protections, all domain controllers must run Server 2008 or later, and the PDC Emulator role must be hosted (at least temporarily) on Server 2012-R2 or later.

Members of the Protected Users group, when they log on locally at Windows 8.1 or Server 2012-R2 or later, enjoy the following protections, but note that these protections can also cause compatibility problems:

- Members must explicitly re-authenticate with NTLM each time NTLM is used because the authentication key computed from the user's password hash is no longer cached in memory.

- Members cannot log on using locally cached credentials, such as when no controller is accessible, because the offline password verifier is no longer retained.

- The Kerberos Ticket-Granting Ticking (TGT) of members cannot be reacquired automatically, the user must re-authenticate manually again when a new TGT is needed.

- Members cannot use CredSSP authentication because plaintext credentials are no longer cached in memory for CredSSP.

- Members cannot use Digest authentication because plaintext credentials are no longer cached in memory for Digest.

Additionally, if all domain controllers are running Server 2012-R2 or later, and the domain functional level is at Server 2012-R2 or later, then the following restrictions are also enforced at the domain controllers for members of the Protected Users group:

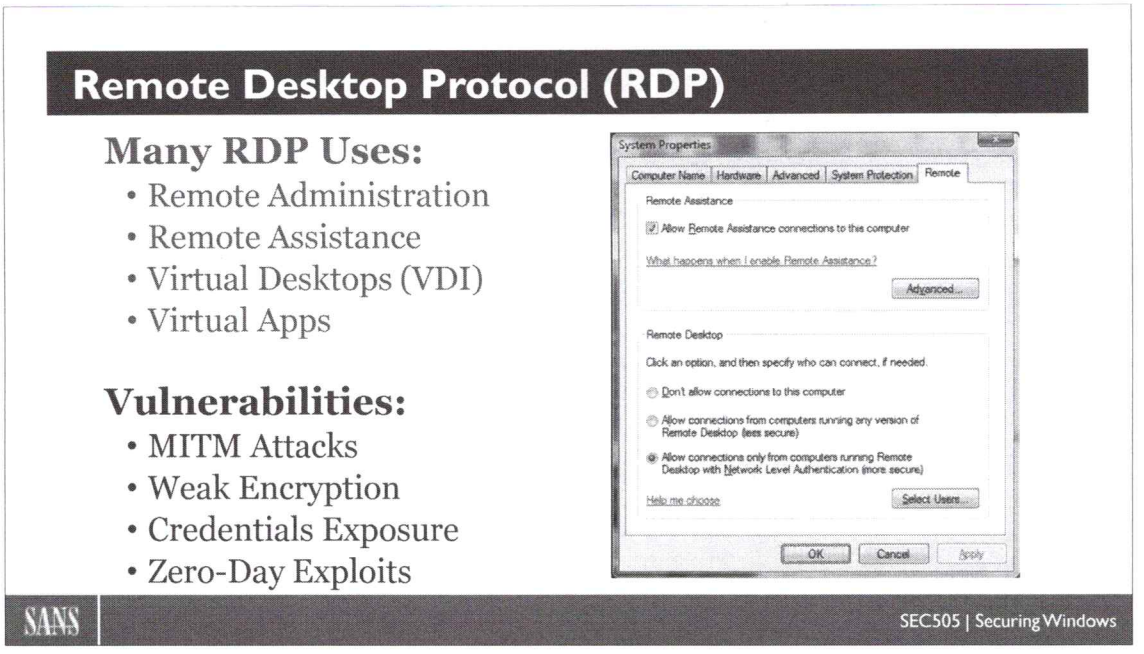
- Members cannot use NTLM at all.

- The Kerberos user tickets of members cannot be renewed beyond the initial 240-minute TTL, requiring a fresh user ticket at least every four hours.

- Members cannot use DES or RC4 for Kerberos.

- Members cannot be trusted for delegation in the properties of a computer account in Active Directory, which is sometimes done for service accounts.

Note that the Protected Users group is not for service accounts or computer accounts. Also, it's recommended that at least one Domain Admins member not be added to the Protected Users group just in case there are problems.



Remote Desktop Protocol (RDP)

Remote Desktop Services (RDS) provides graphical remote control of a hidden desktop running entirely in memory. Users connect to this desktop with Remote Desktop Protocol (RDP) on TCP/UDP/3389. RDP is used for remote administration of servers and to allow users remote access to their own personal computers.

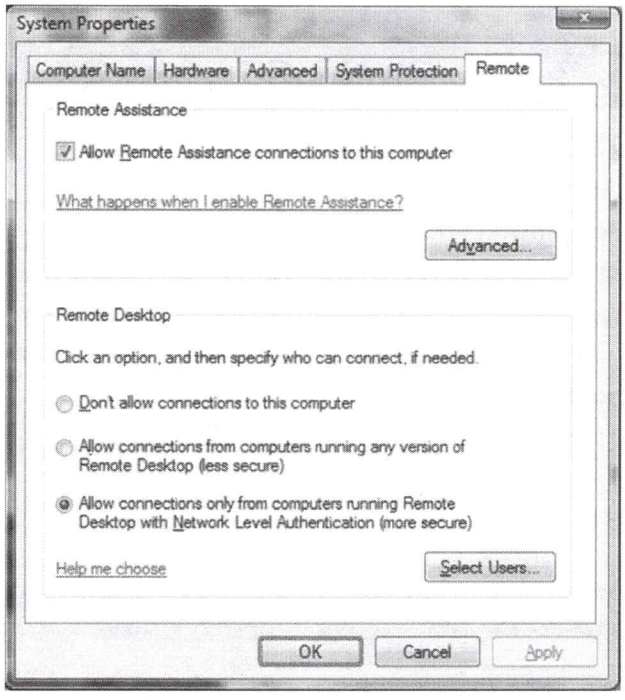
Windows includes a "thin client" application for connecting to remote desktops (MSTSC.EXE), and there are thin clients for Linux, Android, Mac OS, and iOS as well.

Install Remote Desktop Services (Windows Server Only)

When RDS is installed on Server 2012 and later with Server Manager, there is a special installation wizard just for RDS (in Server Manager, pull down the Manage menu > Add Roles and Features > Remote Desktop Services Installation). The wizard is designed to simplify the configuration of RDS across a farm of servers for the sake of a Virtual Desktop Infrastructure (VDI) solution.

Enable Remote Desktop (All Windows Hosts)

On servers and workstations, you can enable or disable Remote Desktop mode without installing the full RDS role by going to Control Panel > System applet > Remote Settings link > Remote tab. A related feature on this tab is Remote Assistance for clients, where a user can allow someone from the Help Desk to remotely view and, with permission, remotely control the user's desktop for troubleshooting or training.



By default, only local Administrators can connect inbound, but in the System applet you can add other accounts too, and you don't have to buy more licenses for this either.

If you are a non-administrative user, you only can connect when either no one is logged on at the remote system or when you are logged on there. If you already are logged on, the remote machine's visible desktop becomes locked. If you are a member of the local Administrators group, you can connect to any machine, but this action will forcibly log off any interactively logged-on user there.

RDS and Remote Desktop both use the Remote Desktop Protocol (RDP) on TCP port 3389 by default. Server 2012 and later also support RDP on UDP port 3389 by default.

Vulnerabilities

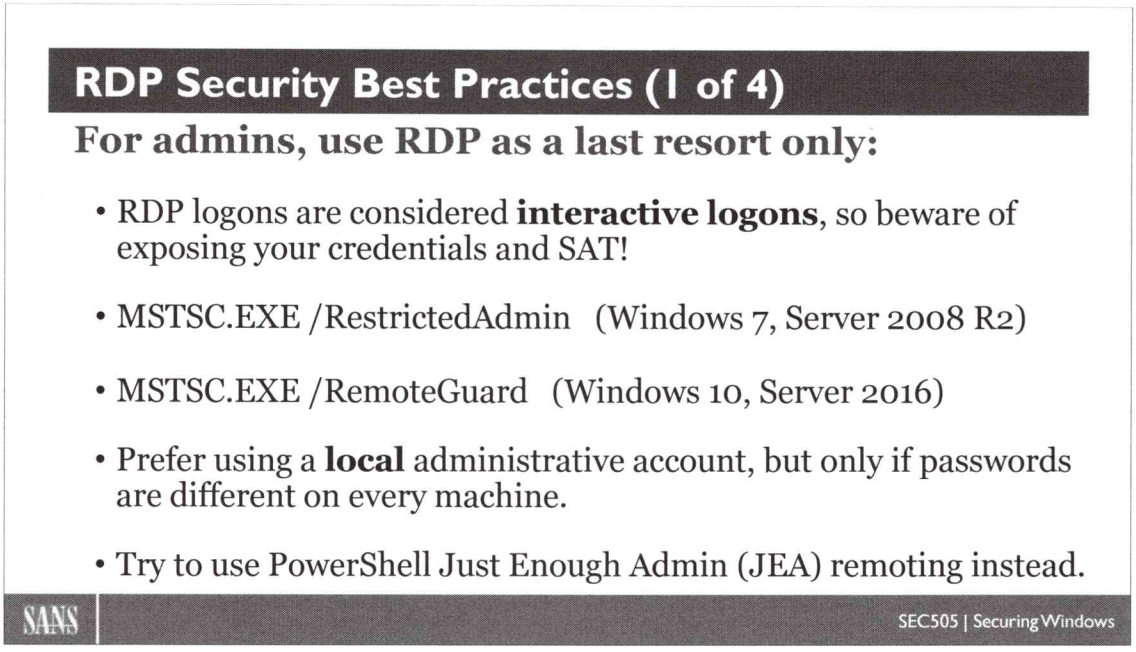
RDP is extremely popular for remote administration, both inside the LAN and over the Internet. As more VMs are pushed up to the cloud, RDP will become increasingly popular and therefore increasingly attacked. Users also RDP into their own computers at home, into their desktops at work, and into VMs for Virtual Desktop Infrastructure (VDI) solutions at work or in the cloud. Literally millions of Internet-accessible IP addresses have their RDP ports directly exposed.

But in its default configuration, RDP is vulnerable to a MITM attack which allows the capture of the user's plaintext password. Self-generated and self-signed certificates are no help because hackers can generate these certificates too. Users habitually ignore any authentication warnings and simply "click through" these warning dialog boxes to get rid of them. Unfortunately, most administrators will do the same thing too.

RDP may also be configured with weak encryption, allowing brute force decryption of the session, which is different than a MITM attack. Administrators may be tempted to use the weaker encryption to maximize performance or for backwards compatibility.

If the RDP target has already been infected with malware, then new RDP connections to that target can reveal more credentials to the malware. From a security perspective, there's not much difference between logging on at the physical console and logging on through RDP. The malware is just waiting for you in either case. This is not a flaw in RDP as a network protocol, but in how Windows handles interactive logon sessions and the caching of credential data in memory.

And, like any service with an exposed listening port, there may be new zero day vulnerabilities published to exploit it. Good patch management is especially important for RDP. RDP zero-day vulnerabilities are quite valuable.



RDP Security Best Practices (1 of 4)

RDS as a service, and RDP as a protocol, are relatively complex. There are several issues to consider for performance, backwards compatibility, user friendliness, PKI integration, VPN gateway usage, IPSec tunneling, and so on. Let's consider some of the issues.

RDP Logons Are Considered Interactive

Even though you might be on the other side of the world, an RDP logon is treated as an *interactive* logon at the target computer (strictly speaking, an RDP logon is logged as a *remote interactive* logon in the event logs). This means your delegation Security Access Token (SAT) and password hash can be stolen by any malware running with administrative privileges on the target.

As a matter of habit, only use RDP as a last resort when you are an administrator. Prefer to use standard remote administration tools instead, such as MMC console snap-ins and PowerShell remoting, and then fall back to RDP only as needed. When you need to use RDP, log on with a local account, if possible, not a global account, and then log off completely when you are done (don't just disconnect, which leaves the session open). Make sure that your local administrative accounts have different passwords on every host, and change these passwords periodically, perhaps weekly.

RDP Restricted Administration Mode (mstsc.exe /restrictedadmin)

On Windows 8.1, Server 2012 R2, and later operating systems, the RDP thin client can be launched with a special command-line switch. This switch is also available for Windows 7, Windows 8, Server 2008 R2 and Server 2012 if the appropriate patches are

installed and the necessary registry values set (see KB2984972 and KB2973501). The special command-line switch launches the thin client in a special mode:

```
mstsc.exe /RestrictedAdmin
```

In this mode, the RDP thin client will not forward a copy of the user's credentials to the remote computer (which is the default with CredSSP authentication), hence, if the remote computer has already been compromised by hackers or malware, these particular credentials will not be available to steal for the sake of pass-the-hash attacks.

This feature requires that both the RDP client and the target computer be running a supported and/or patched operating system, as mentioned above. This is not just a client-side feature of the mstsc.exe tool.

Note that after connecting with RDP to a remote computer in restricted administration mode, when you attempt to connect from that remote computer to a third machine, you will be prompted for credentials. If the remote computer has been compromised, when you enter your credentials again, it's another opportunity for those credentials to be stolen. Hence, when practical, avoid providing your credentials manually after connecting to remote computers with RDP in restricted administration mode; instead, it would be slightly better to directly connect to each target instead of "hopping" from one machine to the next in a chain.

If you wish to force the use of restricted administration mode with RDP on Windows 8.1 and later clients, there is a GPO option for this (GPO > Computer Configuration > Policies > Administrative Templates > System > Credentials Delegation). In this same GPO container you will find other options to control CredSSP credentials sharing and single sign-on, but be careful not to shoot yourself in the foot: at the end of the day, you still have to be able to manage the network even if there are pass-the-hash and SAT abuse risks.

For both restricted admin mode and remote credential guard (below), the following registry value must be set:

Key: HKLM\System\CurrentControlSet\Control\Lsa
Value: DisableRestrictedAdmin
Value Type: DWORD
Value Data: 0

Second Hop Problems

Note that after connecting with RDP to a remote computer in restricted administration mode, when you attempt to connect from that remote computer to a third machine, you will be prompted for credentials. If the remote computer has been compromised, when you enter your credentials again, it's another opportunity for those credentials to be stolen. Hence, when practical, avoid providing your credentials manually after connecting to remote computers with RDP in restricted administration mode; instead, it

would be slightly better to directly connect to each target instead of "hopping" from one machine to the next.

Also, please see KB2973351 and KB2975625 for further complications, because Microsoft has changed what is manageable when using restricted admin mode.

What can be done to make these kinds of "second hop problems" less annoying?

RDP Remote Credential Guard

With Server 2016, Windows 10 version 1607, and later operating systems, there is a new command-line switch for the mstsc.exe tool to help make RDP more secure and less annoying for "second hop" issues:

```
mstsc.exe /RemoteGuard
```

To use the /RemoteGuard command-line switch, the requirements are:

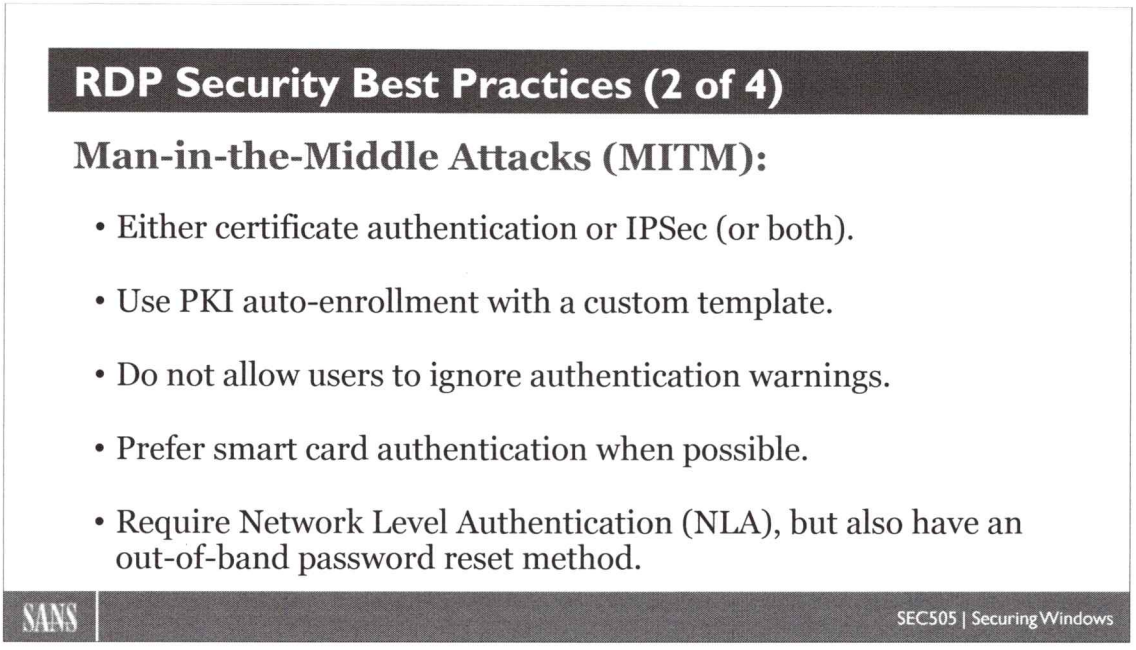
- Both client and RDP server must be running Windows 10 version 1607, Server 2016, or later.
- Neither client nor server can be a stand-alone.
- Neither client nor server can be joined to Azure Active Directory, they must be joined to an on-premises Active Directory domain (or mutually-trusted domains).
- The Remote Desktop Gateway for RDS cannot be used.
- The client cannot provide alternative credentials; single sign-on is mandatory.
- The client must authenticate with Kerberos, not NTLM.

When the above requirements are met, what is the effect of using the /RemoteGuard switch? Just like with RDP restricted admin mode, the user's credentials are not forwarded to the RDP server, hence, these credentials are not held in memory at the (possibly compromised) server either. The difference is that the user's Kerberos authentication to the target server is redirected back to the user's computer for authentication. This means that 1) the user's own computer Kerberos credentials are not being used, as when using the /RestrictedAdmin switch, and 2) the "second hop" problem is solved.

The "second hop" problem is solved because, if the user attempts to connect to a third machine from the RDP server, the Kerberos authentication necessary for this third "hop" is transparently redirected back to the user's computer where the authentication succeeds without the user being prompted for any credentials (it's single sign-on) and none of the user's credentials are exposed to either the RDP server or the third machine being accessed.

This is a major victory for both convenience and security, at least as far as in-memory credentials go, but please don't forget that Security Access Tokens (SATs) are still being created in the memory of all three computers involved (client, RDP server, and third machine) and these SATs may themselves still be abused.

If you wish to enable remote credential guard, or to make it mandatory, or to fall back to RDP restricted administration mode when remote credential guard is not available, then there is an GPO option for this (GPO > Computer Configuration > Policies > Administrative Templates > System > Credentials Delegation, then see the option named "Restrict delegation of credentials to remote servers").



RDP Security Best Practices (2 of 4)

We must also worry about man-in-the-middle (MITM) attacks.

RDP Authentication ("Security Layer")

On Windows Server 2003+SP1 and later you have three choices for authentication of the RDP server (user authentication is another matter):

- **RDP**: Uses native RDP encryption and authentication.
- **Negotiate**: Will try to use TLS, but will fall back to native RDP if necessary.
- **SSL (TLS 1.0)**: TLS is required. Native RDP encryption is not supported.

To set your RDP server's security layer to "SSL (TLS 1.0)", use Group Policy or a direct registry edit (the registry value for this, SecurityLayer, can be set to "2" to require TLS certificate-based authentication).

> ***Try It Now!***
> To configure RDP authentication through Group Policy, edit the relevant GPO > Computer Configuration > Policies > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Security > enable and set the Security Layer option to "SSL (TLS 1.0)".

On Server 2003+SP1 through Server 2008-R2, but not later operating systems, you can also manually make this change by opening the Remote Desktop Services Configuration snap-in in the Administrative Tools folder > Connections > right-click the RDP-Tcp icon > Properties > General tab > select "SSL (TLS 1.0)" for the Security Layer > OK.

Keep in mind that TLS is only available if a digital certificate is available. The Windows PKI supports automatic installation of certificates through Group Policy. Server 2008 and later support the ability to locally auto-generate a certificate if a PKI is not available. In either case, clients will need to support RDP version 5.2 or later.

At the present time, TLS 1.0 is hard-coded. It's not possible to specify TLS 1.1 or later. Also, the ciphers must be RSA and 3DES, not AES, and the hash must be SHA-1.

However, the use of self-generated certificates allows MITM attacks where the attacker simply inserts a certificate generated by the attacker (Cain & Abel can do this for example). Always deploy certificates from your own PKI and do not allow users to ignore authentication errors/warnings and proceed to log onto the server anyway.

> ***Try It Now!***
> To prevent users from ignoring RDP authentication errors and logging on anyway, edit the relevant GPO > Computer Configuration > Policies > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Connection Client > enable and set the Authentication setting option to "Do not connect if authentication fails".

Again, if administrators can ignore authentication warnings, they are vulnerable to MITM attacks. This setting is very important, even more important than the encryption level in fact because MITM attacks are easier and more common.

Whenever possible, require TLS authentication, but do not use self-generated certificates. Instead, deploy certificates through Group Policy from your own PKI and use the GPO setting named "Server Authentication Certificate Template" to specify the name of the certificate template from your PKI which must be used. This will have the effect of preventing the use of self-generated certificates. Make sure to create a duplicate of the existing Computer template, give that template a new name, and ensure that the "Template Name" and "Template Display Name" are exactly identical in that template. If these two names in the template are not identical, then duplicate certificates will be created on machines prior to Windows 8.

If the TLS setting is too harsh, at least use IPSec authentication for RDP. Even stand-alone computers can be quickly configured with an IPSec pre-shared key with a script, if pre-shared key authentication is the only option available (far better than nothing).

RDP Network Level Authentication (NLA)

Windows Vista and later come with the RDP version 6.0 at a minimum, but Windows XP/2003 require a Service Pack or special update to obtain this version (KB925876). It's quite important to upgrade to version 6.0 or later for the sake of security. RDP 6.0+ includes support for Network Level Authentication (NLA), which authenticates the client and server before a session is even created in the memory of the RDP target. NLA helps to prevent DoS attacks against the server and potential credentials-stealing attacks against the client. NLA uses the Credential SSP (CredSSP) provider to give a copy of the user's username and password (not the hash, the password itself) to the RDP server through a

TLS-encrypted channel; this allows a remote interactive logon at the RDP server, but interactive logons also increase the opportunities for malware or hackers to steal the user's credentials. The TLS encrypted channel is authenticated by Kerberos, NTLM, or optionally a certificate from one's own PKI.

To require NLA on Server 2008 or Server 2008 R2, but not later server operating systems, open the Remote Desktop Session Host Configuration snap-in in the Administrative Tools folder > Connections > right-click the RDP-Tcp icon > Properties > General tab > check the box labeled "Allow connections only from computers running Remote Desktop with Network Level Authentication" > OK.

On Server 2012 and later, use Group Policy or a direct registry edit to require NLA (the registry value for this, UserAuthentication, can be set to "1" to require NLA).

> ***Try It Now!***
> To configure RDP Network Level Authentication (NLA) through Group Policy, edit the relevant GPO > Computer Configuration > Policies > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Security > change the "Require user authentication for remote connections by using Network Level Authentication" setting to Enabled.

To require successful NLA on the client's side, open the Remote Desktop Client application > Advanced tab > set the authentication option to "Do not connect." Remember, this requires client version 6.0 or later.

There is a problem with NLA however. If a user's password has expired and must be changed, the user will not be able to change their own password over an RDP connection if NLA is required. In this scenario, the user will need to be able to change their password prior to and independently of any RDP connections, such as through an HTTPS web portal or a phone call to the help desk. A mitigation to this problem is to send e-mail reminders to users prior to their passwords expiring.

Kerberos vs. NTLM

Kerberos is faster and more secure than NTLM. NLA will try to use Kerberos whenever possible instead of NTLM.

But Kerberos cannot be used when:

- The RDP target is a stand-alone computer, not joined to any domain.
- Domain controllers are inaccessible, such as for roaming VPN-less clients.
- RDS farms, because only individual servers can use Kerberos, not whole farms.

We don't want to use NTLM if possible, so in the above scenarios one must use certificates from the PKI and prevent users from authorizing any new certificates or CAs in their client applications. Without these constraints the servers won't be authenticated and users will be vulnerable to MITM attacks. When a PKI is not available, which is especially common on stand-alones, the next best option is to somewhat give up on RDP

handling its own security and instead require IPSec for all RDP traffic. Indeed, you might conclude from this whole section that IPSec should be used for RDP whenever possible; IPSec might be easier to manage than the PKI and NLA issues.

If Kerberos and certificate authentication both fail for any reason, then the RDP client will fall back down to NTLM by default. It's best not to use NTLM, so NTLM should be disabled globally (best) or at least disabled for use with CredSSP, NLA and RDP (next best). To disable NTLM for RDP or anything else which uses CredSSP, go to the appropriate GPO > Computer Configuration > Policies > Administrative Templates > System > Credentials Delegation, then disable the following settings:

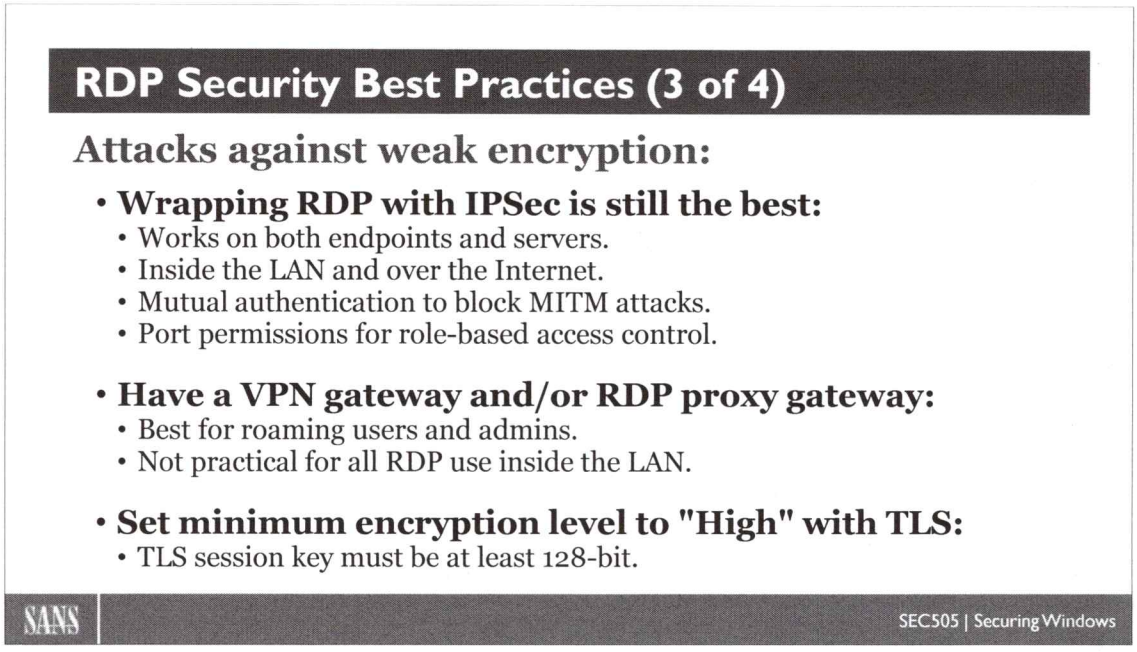
- Allow Default Credentials with NTLM-only Server Authentication
- Allow Fresh Credentials with NTLM-only Server Authentication
- Allow Saved Credentials with NTLM-only Server Authentication

Again, if NTLM is disabled for RDP, then you must use either Kerberos or certificate authentication. Kerberos cannot be used on stand-alones, it's a pain to install and manage certificates on stand-alones (can't use auto-enrollment), so it's either NTLM and/or IPSec.

Be aware, too, that the above GPO settings affect not just RDP, but any protocol or network service which uses CredSSP authentication. Beyond RDP, the use of CredSSP is uncommon, but it's not totally rare; PowerShell remoting has the option of using CredSSP, for example, even though it is not enabled by default.

RDP Single Sign-On

In Windows Vista, Server 2008 and later, RDP supports both NLA and domain account single sign-on; hence, after an administrator logs onto her desktop with a global user account, that administrator's RDP client can transparently authenticate to the RDP server without re-entering a password or smart card PIN number. This transparent authentication occurs after the NLA authentication, but the client will have to be Vista or later to make it all work seamlessly.



RDP Security Best Practices (3 of 4)

If RDP uses weak encryption, the keys can be cracked by brute force or flaws in the cryptographic implementation. There are multiple options for RDP encryption.

RDP Encryption Level

On Windows Server 2003+SP1 and later, there are four possible encryption levels:

- **Low**: Client determines encryption strength and only data sent to the server is encrypted. Data received from the server is plaintext.

- **Client Compatible**: Encryption strength is determined by the client, but all data is now encrypted, both to and from the server. TLS is permitted if requested.

- **High**: Encryption is set to the server's highest level of encryption possible, but it will be at least 128-bit RC4 when using RDP native encryption. Any clients that cannot support it will be rejected. TLS encryption is permitted if requested by the client.

- **FIPS Compliant**: Algorithms used must be 3DES, RSA and SHA (not even AES). RC4 is not permitted either. TLS is permitted if requested.

If TLS encryption is used, the packets do not use TCP port 443. The TLS-encrypted traffic still uses TCP 3389, but that channel's payload is now encrypted differently. If you wish to tunnel RDP over HTTPS on TCP port 443, install the Remote Desktop Services Gateway role on Server 2008-R2 or later. If you wish to simply use TCP port 443 instead, the RDP port number can be changed to 443, as long as no other service

needs to use port 443 on that server. Changing the default port number is a simple registry edit.

To set your server's encryption level to High, use Group Policy or a direct registry edit (the registry value for this, MinEncryptionLevel, can be set to "4" to require FIPS-compliant TLS encryption).

> ***Try It Now!***
> To configure RDP encryption through Group Policy, edit the relevant GPO > Computer Configuration > Policies > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Security > enable and set the encryption level to "High Level".

On Windows Server 2008-R2 and earlier, you can also make this change manually by opening the Remote Desktop Session Host Configuration snap-in in the Administrative Tools folder > Connections > right-click the RDP-Tcp icon > Properties > General tab > select High for the encryption level > OK.

But IPSec Is Still The Best For Security

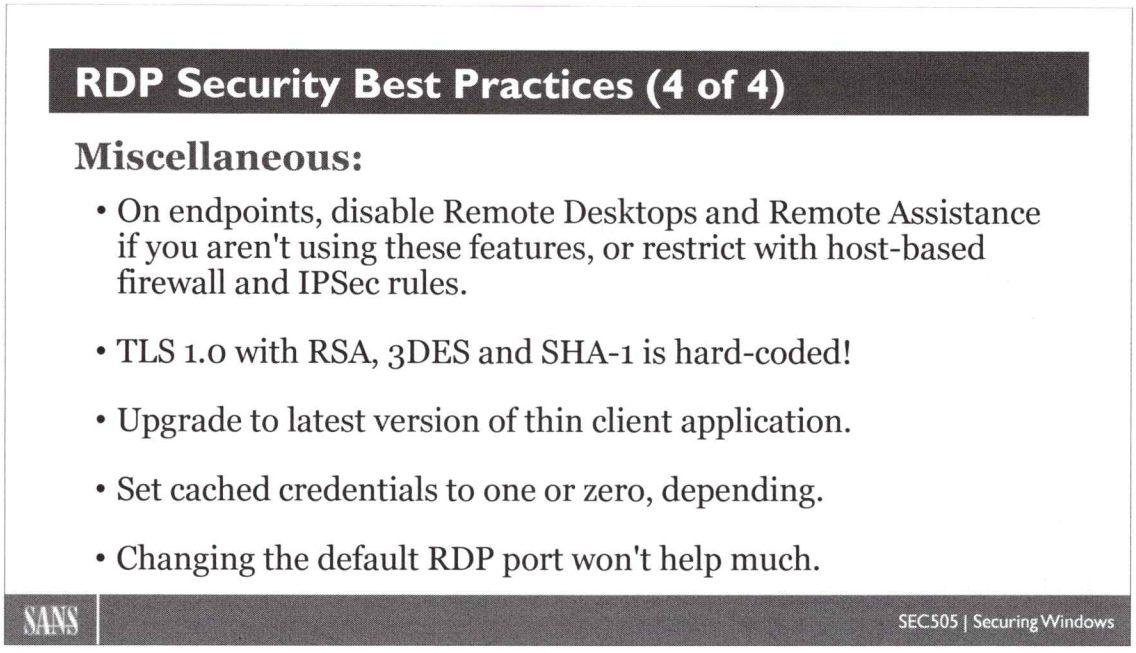
Even with TLS and the minimum encryption level set to "High", IPSec is still better. IPSec provides mutual authentication and modern cipher suites, such as 256-bit AES, elliptic curve Diffie-Hellman, and SHA-256. IPSec works on both client endpoints and the target servers. IPSec can be managed through Group Policy and PowerShell. IPSec can also provide TCP/UDP port permissions in order to limit access to these ports based on group memberships in Active Directory for role-based access control.

However, IPSec is sometimes still blocked at hotels, coffee shops, airports and other locations where users find themselves roaming. In this case, we will need some kind of gateway for RDP.

VPN Gateway And/Or RDP Proxy

For the sake of roaming users, RDP services can be accessed through a VPN gateway. The VPN gateway can require multi-factor authentication and strong encryption. Through the VPN, users can still use IPSec to wrap their RDP traffic.

Alternatively, there are RDP-compatible proxy servers from Microsoft, Citrix and other vendors. These proxies wrap the RDP traffic inside an encrypted HTTPS layer. HTTPS is virtually always allowed outbound on the networks of roaming users, such as at airports, coffee shops, and hotels. The RDP proxy can also require various forms of authentication, depending on the vendor.



RDP Security Best Practices (4 of 4)

There are other security considerations for RDP, though they are less important. Not listed in the slide is patch management, which is extremely important, but also taken for granted in a course like this.

Principle of Least Endpoint

If you do not plan to use Remote Desktops or Remote Assistance on user endpoints, then disable these features through Group Policy or your hardening templates. If you want to keep these features, then restrict access only to the group(s) which need it, and use host-based firewall and IPSec rules to restrict access.

Consider blocking access to local hard drives, the clipboard and PnP devices (except smart cards) from within the RDP client. These are potential malware vectors, especially for local hard drive access.

Use Group Policy to lock down the virtual desktops of thin client users. Virtually every aspect of the user's desktop can be locked down through GPO.

If you allow users to send Remote Assistance invitations, set the invitation expiration timer to a relatively short value (3 to 24 hours) and require a passphrase to connect. These options also can be configured through Group Policy.

TLS 1.0 with RSA, 3DES and SHA-1?

TLS is only available if a digital certificate is available. At the present time, TLS 1.0 is hard-coded. It's not possible to specify TLS 1.1 or later. Also, the ciphers must be RSA

and 3DES, not AES, and the hash must be SHA-1. It is unknown when or whether Microsoft will change this. This is another reason to consider using IPSec.

Locally Cached Credentials

RDP logons also leave behind locally cached credentials. The password verifier in locally cached credentials is difficult to crack, especially when users have good passphrases, but ideally we don't want this information exposed anyway. It's best to set the count of locally cached credentials to either zero or one at the target computer, depending on where it is located and what type of machine it is.

On mobile computers, the count of cached credentials will have to be set to at least one for when the computer is outside the LAN without a VPN. On internal workstations, when it is likely a user will log on again soon, it can be set to one if domain controller access is not assured; then, if an administrator logs on with RDP, the odds of a user logging on soon are good, and then the administrator's locally cached credentials will be overwritten with the user's. The key thing is the availability of a domain controller over a reliable network.

On internal servers, though, sometimes the only interactive logons are the RDP sessions of administrators, so on these machines it is best for security to set it to zero (and then have multiple, reliable domain controllers). If no domain controllers are available and the count of cached credentials has been set to zero, no one will be able to log on to the computer with a domain account, so there is a DoS risk if there are no local accounts available. Each server will have at least one local administrative account, and the passwords of these accounts across servers should be different and periodically changed.

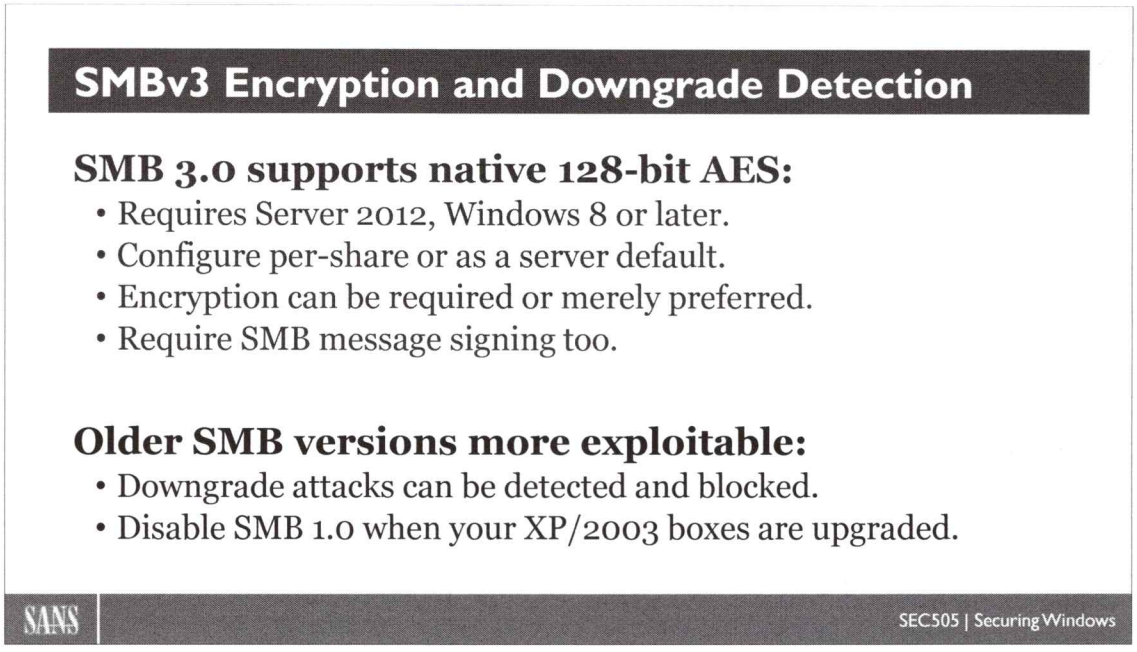
The GPO setting for cached credentials is located in a GPO under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options > Interactive logons: Number of previous logons to cache (in case a domain controller is not available).

RDP Port Number

RDS and Remote Desktop both use the Remote Desktop Protocol (RDP) on TCP port 3389 be default. Server 2012 and later also supports RDP on UDP port 3389 by default.

If you wish to change the TCP port, edit the PortNumber value located underneath HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp.

This might help against automated worms which have a hard-coded port number, but a hacker performing full port scans will quickly find the new port. At a minimum, choose a port number which is not scanned by Nmap by default, e.g., 48777 (www.nmap.org).



SMBv3 Encryption and Downgrade Detection

The Server Message Block (SMB) protocol is used to access shared folders, shared printers, and RPC services which opt to use SMB as their underlying transport. With NetBIOS session management, SMB operates on TCP/139, but when NetBIOS is not used, SMB operates on TCP/445. NetBIOS is not required for SMB.

There are many "dialects" or versions of SMB, each with different security and performance characteristics. Server 2012 and Windows 8 introduced a new version (SMB 3.0) but the SMB version actually used between any two machines is negotiated.

Version	Minimum Required Operating System
SMB 1.0	Windows 2000, XP, Server 2003, Server 2003-R2
SMB 2.0	Windows Vista+SP1, Server 2008
SMB 2.1	Windows 7, Server 2008 R2
SMB 3.0	Windows 8, Server 2012, Linux/Unix Samba 4.1
SMB 3.0.2	Windows 8.1, Server 2012 R2
SMB 3.1.1	Windows 10, Server 2016

To see what version of SMB is currently being used to access a shared folder:

```
Get-SmbConnection      #Requires Server 2012, Windows 8 or later.
```

SMB 3.0+ includes enhancements for transparent failover, multi-channel I/O, Remote Direct Memory Access (RDMA) on supported NICs, and, most importantly for this course, native AES encryption. On Server 2012, Windows 8 and later, it is also possible to simply turn off support for SMB 1.0 to reduce future vectors of attack.

Historically, SMB has been a favorite target of hackers and there are many published SMB vulnerabilities (for examples, see MS08-067, MS04-011, MS06-040 and MS10-061). Millions of computers have been compromised using SMB exploits in the past, so we must expect future attacks as well. It is important to secure SMB traffic and listening ports even if you have a perimeter firewall.

Require SMB Encryption

SMB encryption can be enabled globally on the server or on a per-share basis. The cipher with SMB 3.1.1 is 128-bit AES-GCM or AES-CCM (GCM is preferred), the signing method is 128-bit AES-CMAC, and the initial negotiation uses an SHA-512 hash. (With SMB 3.0, the cipher is only 128-bit AES-CCM, but this only about half as fast as AES in GCM mode, hence, use SMB 3.1.1 or later to optimize performance.)

To require SMB encryption for the entire server:

```
Set-SmbServerConfiguration -EncryptData $True
```

To require SMB encryption for one shared folder:

```
Set-SmbShare -Name <ShareName> -EncryptData $True
```

SMB encryption can also be enabled with Server Manager (File and Storage Services > Shares > right-click the desired share > Properties > Settings > check "Encrypt data access" > OK).

To list which shared folders do or do not require SMB encryption:

```
Get-SmbShare | Format-Table Name,Path,EncryptData -AutoSize
```

Prefer SMB Encryption

By default, when a server or share is marked for SMB encryption, the encryption is mandatory and only clients capable of SMB 3.0 or later (Windows 8, Server 2012 or later) will be able to connect to the share. This requirement may be too harsh. It may be feasible only to *prefer* SMB encryption instead of requiring SMB encryption until all clients have been upgraded.

To change the global policy of the server to only prefer, not require, SMB encryption:

```
Set-SmbServerConfiguration -RejectUnencryptedAccess $False
```

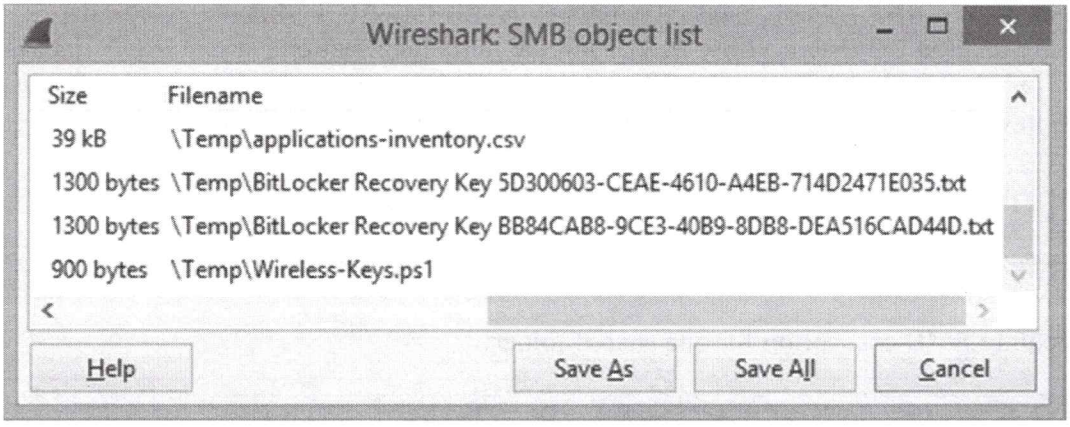
> **Note:** Server-wide SMB encryption settings are stored in the registry under HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters.

It is not possible to configure prefer-versus-require mode on a per-share basis.

The minimum version of Linux/Unix Samba to support SMB encryption is 4.1.0.

Example Attack: Wireshark SMB File Extraction

It is trivially easy to sniff SMB packets with Wireshark and extract files from captured SMB sessions (www.wireshark.org). In Wireshark, pull down the File menu > Export Objects > SMB > select the file(s) you want > click Save As. An attacker could sniff SMB packets on a compromised computer, perhaps with windump.exe or tshark.exe, saving the packets to a large PCAP file, later uploading that file to an Internet location that the attacker can access, then perform the SMB file extraction offline with Wireshark.



Downgrade Attack Detection

SMB 3.0 attempts to detect attacks which try to downgrade the SMB version. Attackers might wish to do this when an exploit only works with an older version of SMB.

When a downgrade attempt is detected, the entire SMB session is halted and event ID 1005 is written to the Microsoft-Windows-SmbServer > Operational event log. This is called "secure dialect negotiation", but the details are not published (yet), so there's no guarantee that hackers won't find a method to defeat it.

However, be aware that this feature can only detect and block downgrade attacks down to version 2.0 or 2.1, it cannot prevent downgrades all the way down to SMB 1.0.

If a server or shared folder is SMB-encrypted, but the encryption is not mandatory, then an attack could downgrade an SMB session from 3.0 to 1.0, which would disable the encryption and potentially expose the server to exploits affecting SMB 1.0.

Hence, if you want to always enforce SMB encryption, you must also disable support for SMB 1.0 entirely, which will block the downgrade attack.

Disable SMB 1.0 (When All Systems Run Vista/Server 2008 or Later)

SMB 1.0 has a history of problems and exploits. The protocol dates back far prior to the entire Trustworthy Computing Initiative at Microsoft. When possible, disable SMB 1.0 and retire it forever to reduce your exposure to SMB attacks.

Server 2008, Windows Vista+SP1 and later operating systems support the disabling of SMB 1.0. There will never be an update for Windows 2000/XP or Server 2003/2003-R2 which allows disabling SMB 1.0 on those older platforms.

To disable SMB 1.0 on Server 2012, Windows 8 and later using PowerShell:

```
Set-SmbServerConfiguration -EnableSMB1Protocol $False
```

To disable SMB 1.0 on Server 2008, Windows Vista or later:

```
Set-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters"
-Name SMB1 -Type DWORD -Value 0 -Force
```

The above registry value can be set using an INF template, exported REG file, or other technique as well; it does not have to be set using PowerShell. Set the value back to 1 to re-enable SMB 1.0.

You will only be able to disable SMB 1.0 on a server if all of its clients have been upgraded to at least Windows Vista+SP1 or Server 2008.

> **Warning!** Do not disable SMB 2.0 or else SMB 3.0 will also be disabled. This is a bug which may or may not be patched at a later date.

It is also possible to disable support for SMB 1.0 for the computer when it is acting as an SMB client or "redirector" using the Workstation service (see KB2696547).

If an SMB 1.0 client is denied access to a server because SMB 1.0 has been disabled, event ID 1001 will be logged to the Microsoft-Windows-SmbServer > Operational event log on the server. The client's name and IP address are recorded as well.

SMB Encryption Details

The 128-bit AES keys used for encryption and signing are ultimately generated from seed material produced by the user's initial Kerberos or NTLM authentication to a domain controller. If the initial interactive logon is relatively secure, then, following the long chain of key derivation functions needed to produce the SMB keys, the resulting keys are relatively secure too.

Why the "relatively" weasel-word in the prior sentence? Because if a user authenticates with a smart card, the initial logon is about as secure as it gets on Windows, so when SMB needs an encryption key later on, that AES key is also about as good as it gets on Windows too, which in this specific case, is very, very good. But if the user authenticates to the domain with a blank password and NTLMv1, then the resulting SMB encryption is horrible; and not horrible because the key is not 128 bits long, it still is, it's horrible because the entropy or randomness of that key is horrible (the "bits of entropy" in this key would be nearly zero).

Note that Server 2012 and Windows 8.1 use SMB 3.0 with AES 128-bit in a special mode of operation: Counter with CBC-MAC (CCM) mode. Server 2016, Windows 10 and later, on the other hand, use AES 128-bit with Galois Counter Mode (GCM) with SMB 3.1.1. The switch from AES-CCM to AES-GCM improves throughput by about 100%. SMB 3.1.1 also includes a pre-authentication check using SHA-512 to defeat tampering attacks during SMB session establishment. For SMB encryption, then, try to use Windows 10, Server 2016, or later to optimize performance.

If you'd like to dive into the details, start with these Microsoft documents, beginning with the first and following the trail of references:

- "Server Message Block (SMB) Protocol Versions 2 and 3 Specification", last seen at http://msdn.microsoft.com/en-us/library/cc246482%28v=prot.13%29.aspx

- "[MS-KILE]: Kerberos Protocol Extensions", specifically section 3.1.1.2, last seen at http://msdn.microsoft.com/en-us/library/cc233883%28v=prot.13%29

- "[MS-SPNG]: Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) Extension", last seen at http://msdn.microsoft.com/en-us/library/cc247021%28v=prot.13%29

SMB Message Signing

Windows has supported digital signatures for SMB traffic for a long time (KB887429). SMB signing helps to defeat some types of man-in-the-middle attacks, such as the authentication reflection attack against SMB sessions, but not all (see UNC Hardened Access just below).

SMB signing has been enabled, but not required, for *outbound* SMB sessions ever since Windows NT 4.0 SP3, and has been required for *inbound* sessions to domain controllers ever since Server 2003. Other than domain controllers, though, nothing else requires SMB signing for inbound connections; this is done for the sake of widest compatibility, but at the price of security, as usual.

If you have control over all the computers in your environment, both Windows and non-Windows, then you can safely require SMB signing for all SMB traffic. The only negative is the performance overhead from the extra CPU cycles required for the signing, which can be as high as 15%.

But if you do not have control over all the computers which may need to use SMB, or if you wish to gradually ease your way towards 100% signed SMB traffic, then start with your high-value targets and configure just them to require inbound SMB signatures too. This is already the case on domain controllers running Server 2003 or later, so adding the same requirement to other critical workstations and servers should not be especially risky. The main risk comes from BYOD computers and older non-Windows devices.

To configure SMB signing for either inbound or outbound connections, and to make signing either mandatory or optional, navigate in a GPO to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options:

- Microsoft network **client**: Digitally sign communications (always)
- Microsoft network **client**: Digitally sign communications (if server agrees)
- Microsoft network **server**: Digitally sign communications (always)
- Microsoft network **server**: Digitally sign communications (if client agrees)

Keep in mind that all Windows computers can act as both an SMB server and an SMB client simultaneously; these terms, "client" and "server", do not refer to operating system editions, they only refer to the role(s) of the computer at the moment the policy is enforced for the sake of SMB.

Importantly, note that SMB 3.1.1 introduced a pre-authentication check using SHA-512 to defeat tampering attacks during SMB session establishment. This pre-authentication check, when combined with SMB encryption, supersedes traditional SMB signing, but it requires Windows 10, Server 2016 or later on both sides of the SMB session.

UNC Hardened Access

Unfortunately, Microsoft's original implementation of SMB signing was flawed and not all man-in-the-middle attacks were blocked (see KB3000483). With updates numbered 3004375 and 3000483 both applied, Windows Vista and later computers support a feature named "UNC Hardened Access" (but not Server 2003 or Windows XP). UNC Hardened Access can be configured through Group Policy to require additional mutual authentication for selected UNC paths, and these paths may include wildcards in the server name and share name portions of the paths. UNC Hardened Access can also use these defined paths to require SMB signing and encryption as described above, assuming the client supports SMB encryption (only Windows 8, Server 2012, and later).

Mutual authentication is already provided by Kerberos, but UNC Hardened Access extends that authentication to SMB after Kerberos has been utilized at the beginning of the SMB connection. In short, mutual authentication is incorporated into every SMB request and response when UNC Hardened Access is enabled.

With the two patches above applied to all affected clients and servers, clients can be configured through Group Policy to require mutual authentication, integrity checking and encryption for UNC paths that the administrators select; for example, the following are valid strings to define UNC paths that may be added to a GPO:

- \\server\share (one specific shared folder on one server)
- *\share (one specific shared folder on any server)
- \\server* (all shared folders on a specific server)

The GPO setting is named "Hardened UNC Paths" and is located in the GPO under Computer Configuration > Policies > Administrative Templates > Network > Network Provider.

The recommendation is to apply the above patches to all clients and servers, then require at least SMB mutual authentication for "*\NETLOGON" and "*\SYSVOL", which are shared folders found on domain controllers (see security bulletin MS15-011). A good write up of the problem can be found by searching "site:blogs.technet.com ms15-011 ms15-014 hardening group policy".

Best Practices

Here are the best practices for benefiting from SMB 3.0 and later:

- Upgrade to Server 2012, Windows 8 or later operating systems, at a minimum, to benefit from SMB encryption. Upgrade to at least Server 2016, Windows 10 or later to benefit from the performance gains of AES-GCM.

- When feasible, require SMB encryption by default for all servers, or at least on critical shared folders. During the upgrade transition, it may only be feasible to make the encryption preferred, not mandatory. Alternatively, IPSec can be required for sensitive SMB traffic during the upgrade transition. IPSec is backwards compatible to Windows 2000.

- When possible, disable SMB 1.0 on Server 2008, Windows Vista+SP1 and later systems. Clients will then need to be at least Vista+SP1, Server 2008, or later.

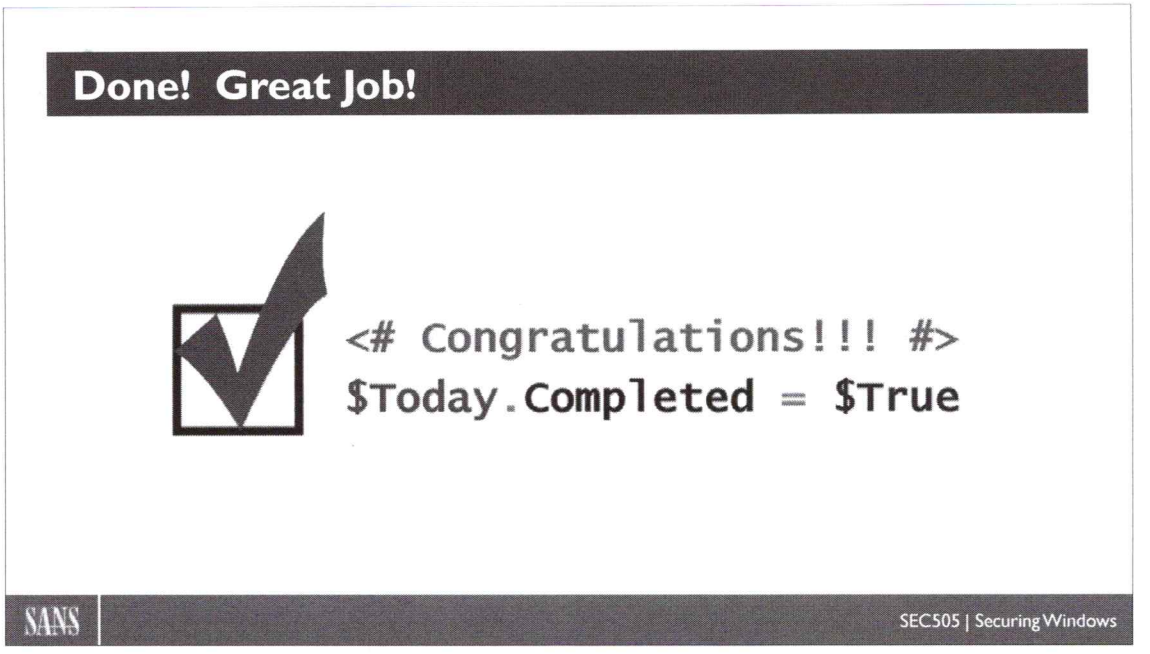
- Require smart card authentication, or at least a long passphrase.

- Disable NTLM when possible and only use Kerberos, preferably with Kerberos armoring enabled.

- Leave SMB signing enabled on all systems, but require SMB signing on high-value targets, such as Domain Admin laptops and mission-critical file servers. If you control all the computers using SMB, then make SMB signing mandatory on every computer, but beware of BYOD computers and older non-Windows boxes.

- When security patches are released that affect SMB, Kerberos or NTLM, test and apply them as quickly as practical. These vulnerabilities will be actively exploited.

- Configure UNC hardened paths for *\NETLOGON and *\SYSVOL at a minimum, with more servers and shared folders as needed.

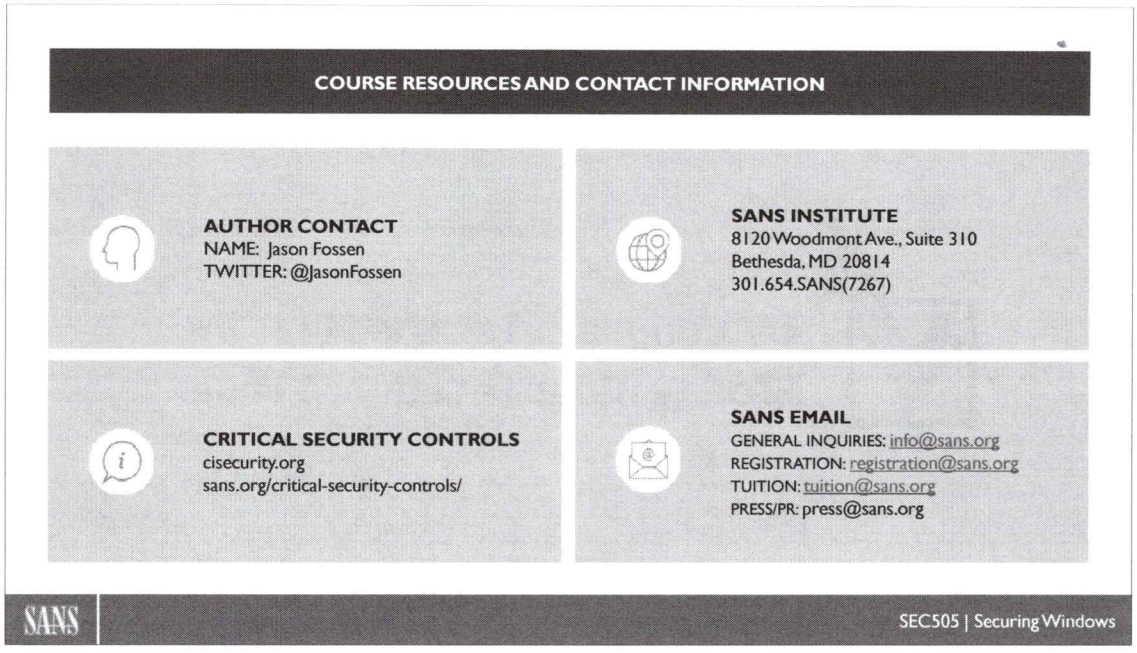


Congratulations!

You have completed the course!

Thank you for attending this seminar, and please remember to complete the evaluation form -- your input decides how the course will be updated in the future.

Best Wishes and Good Luck!



This page intentionally left blank.

Appendix A: Restricting Anonymous Access

Anonymous access in various forms, especially "null user session" access, has been a thorn in Microsoft's side for years. For example, with a null user session you can extract a list of all domain user accounts. But what are null user sessions and how can they be restricted?

What Is A "Null Session User"?

When a user attempts to access data or services on a remote system and authenticates with NTLM, then, even if the username or password is rejected, *some* applications will automatically attempt to establish a connection where both the username and password are each a single null character. This is especially true for SMB-based applications.

A null session is created when a client logs into a Windows host and both the username and password are null characters. When a null session user logs on, that user is represented by the Anonymous Logon account in Windows Server 2003 and later (well-known SID = S-1-5-7). The Anonymous Logon account is not a member of the Authenticated Users or Everyone groups by default, but it can be made a member of the Everyone group by enabling the "Network access: Let Everyone permissions apply to anonymous users" GPO option (see below).

What It Is Not

Human users do not enter null characters for usernames and passwords; this functionality is simply built into various applications and services. Nor is there an account named "null" in the AD or local account databases. The null user is also not the same as the anonymous user account for Internet Information Services (IUSR). Nor are null user sessions identical to automatic Guest logons when one's username is completely unknown to the server.

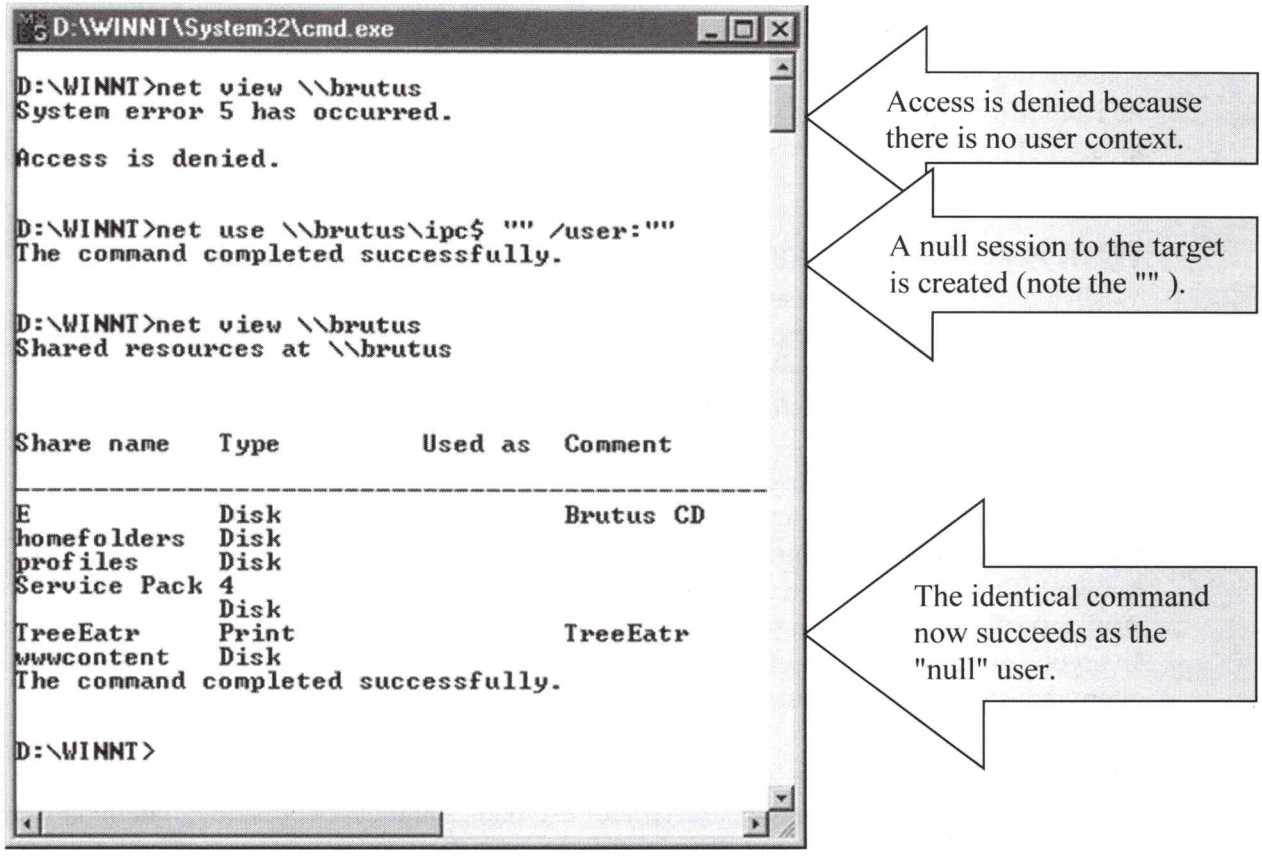
It was widely believed that null user sessions are vulnerabilities of the NetBIOS protocol, but this is not true. NetBIOS can be disabled and null user sessions are still possible.

Null sessions are mainly used for administrative purposes where one's user account is unavailable or lacks sufficient rights/permissions. It is also used by certain network services when they communicate across the network. The functionality of null sessions is limited, but even this limited functionality has opened security holes. Fortunately, it is possible to further limit the power of null sessions through configuration of the registry.

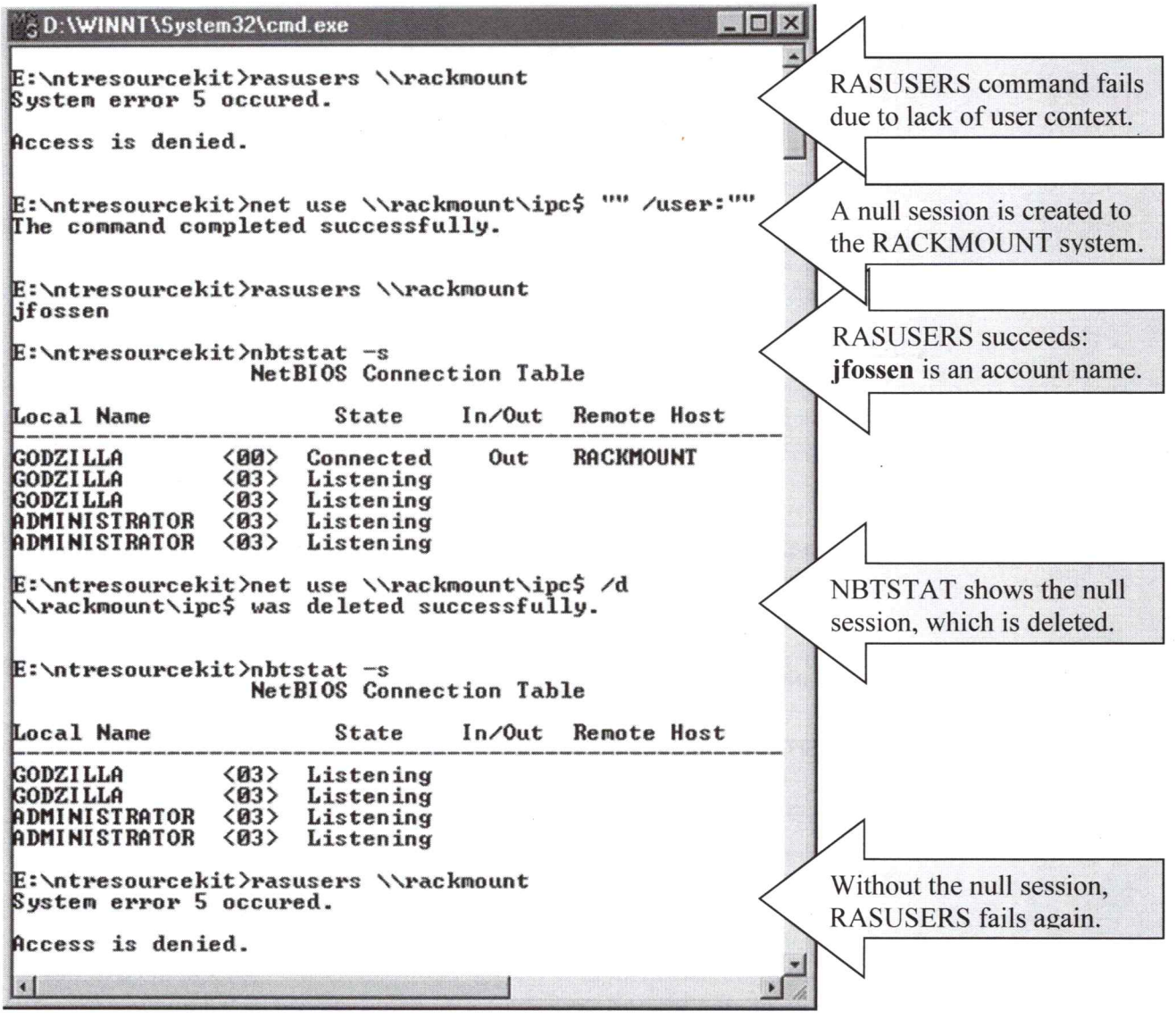
Examples

In the screenshot below, a hacker attempts to list the sharenames on a target computer named BRUTUS. The attempt fails because the hacker does not have a user context on the target. An SMB connection is made to the target (net use \\brutus\ipc$ "" /user:"") with

no characters sent for either the password or username. When the hacker attempts to list sharenames again, she is successful.



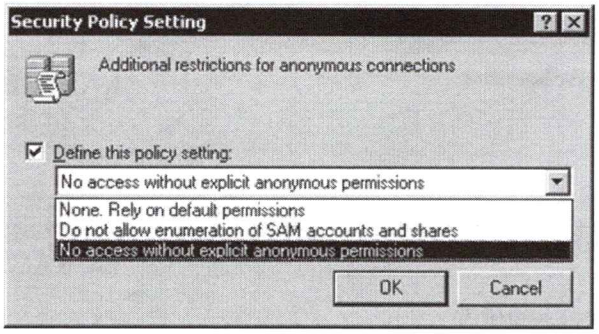
Consider another example using the *Resource Kit* utility RASUSERS.EXE, which will list all user accounts which have the dial-in permission (e.g., JFOSSEN below). The NBTSTAT.EXE -S command lists NetBios sessions on the local machine. (If NetBIOS is disabled you won't see much.)



A null session can also be used to list all users and groups in a target domain from a remote computer. There are very many tools for this, such as BindView's ENUM.EXE (razor.bindview.com).

GPO Settings To Restrict Anonymous Access

There are a number of Group Policy settings related to anonymous access. Unless otherwise specified, they're located under Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options.



Additional Restrictions For Anonymous Connections
On Windows 2000, the default permissions allow enumeration of usernames, sharenames, password policy, lockout policy, and other information through anonymous connections. This setting has three options in order of increasing security:

- **None.** This is the default. Permissions are left alone and your machine will behave as though it were Windows NT 4.0. Sometimes this option is necessary for backwards compatibility with older or third-party software.

- **Do not allow enumeration of SAM accounts and shares.** This also prevents a listing of AD accounts, despite the SAM reference. This option behaves identically to the RestrictAnonymous registry setting on Windows NT. This blocks many null session attacks while breaking as little as possible; but note that there are other attacks that still work despite this.

- **No access without explicit anonymous permissions.** There is a built-in system group named Anonymous Users (later renamed the Anonymous Logon account in Windows Server 2003). When this option is selected, anonymous users will only get the permissions and rights granted to the Anonymous Users group. This should always be selected unless a backwards-compatibility issue demands otherwise. Strictly speaking, this option removes the SID of the Everyone group from the Security Access Token (SAT) of anonymous users, hence, the Anonymous Logon account is effectively removed from the Everyone group.

Windows XP and later has broken out this option into separate items, but the intent is still the same even if the registry value(s) have changed:

Network Access: Do not allow anonymous enumeration of SAM accounts.
Recommendation: Enabled.

Network Access: Do not allow anonymous enumeration of SAM accounts and shares.
Recommendation: Enabled.

Network Access: Let Everyone permissions apply to anonymous users.
Recommendation: Disabled.

Network Access: Restrict anonymous access to named pipes and shares.
Recommendation: Enabled.

Network Access: Named pipes that can be accessed anonymously.
Recommendation: Ideally this should be blank, but you may have applications or services that require a particular value. Testing is required.

Network Access: Remotely accessible registry paths and subpaths.
Recommendation: Ideally this should be blank, but you may have applications or services that require a particular path. Testing is required.

Network Access: Shares that can be accessed anonymously.
Recommendation: Ideally this should be blank, but you may have applications or services that require a particular share. Testing is required.

Network Access: Allow Anonymous SID/Name Translation.
If enabled, this option permits anonymous users to determine the usernames of accounts with well-known SIDs. This means a renamed Administrator account can be discovered. Moreover, with the correct "SID walking" tool a hacker can still extract all usernames even if enumeration of accounts is being blocked (see UserInfo and UserDump from www.hammerofgod.com, and dumpsam from razor.bindview.com). Hence, always disable this option. Unfortunately, it only exists on Windows XP and later. Recommendation: Disabled.

Because null sessions are used by some (ancient) network services, these services may break when the above settings are enabled. Also, when two domains only have a one-way explicit trust between (which is not the same thing as a cross-forest trust), users in the trusting domain will not be able to enumerate user accounts from the trusted domain, such as when assigning NTFS permissions to accounts in the trusted domain.

Other Enumeration Attacks

Since null user sessions are often associated with enumeration attacks, it's worth mentioning that Active Directory permissions should also be hardened to defend against anonymous enumeration of objects. Specifically, the Everyone group and the Anonymous Logon account should be removed from a built-in group named "Pre-Windows 2000 Compatible Access". This may break features on older operating systems and software, but it's better to upgrade these dinosaurs than to hope.