

Use offense to inform defense.
Find flaws before the bad guys do.

Copyright SANS Institute
Author Retains Full Rights

This paper is from the SANS Penetration Testing site. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, Exploits, and Incident Handling (SEC504)"
at <https://pen-testing.sans.org/events/>

ITDOCS

One Admin's Documentation is their Hacker's Pentest

GIAC GPEN Gold Certification

Author: Rob VandenBrink, rvandenbrink@metafore.ca

Advisor: Pedro Bueno

Accepted: 11 December, 2009

Abstract

At the beginning of a recent client engagement, I was asked to create system documentation for a mixed Windows, Linux and Netware environment. Since the actual engagement was to include upgrades to all of these components, documenting the soon-to-be-destroyed environment didn't seem like time well spent. Instead, it was decided to automate the documentation, using a set of scripts with a simple front-end web page. As the scripts were created, it was found that the initial job of documentation actually took significantly less time than assembling and arranging the system information manually, even accounting for the development effort. Since the documentation was automated, we scheduled it to run daily, ensuring that system documentation was always up to date during the project. This set of scripts has grown to a point that they are worth describing in their own right. The collection of scripts has been named "IT-DOCS", since "IT" Documents "IT" infrastructure. IT-DOCS now includes scripts to generate documentation reports for various platforms, scripts to audit for change control and scripts to audit for compliance.

1. Introduction

This paper describes the background, design and specifics of an automated, script based documentation system for IT infrastructure called IT-DOCS. We begin with a short background on scripting, outlining common motives for scripting – for simplifying common, repetitive operations, to facilitate repeatability for benchmarks and audits, and for training and education purposes. It is felt that the IT-DOCS project fills all three of these goals.

We then discuss specific uses for scripting that are addressed by IT-DOCS – namely documentation, audit for compliance and audit for change control. Operating System specific scripting methods and issues are discussed, as well as how scripting, documentation and IT-DOCS interacts with penetration tests.

Finally, specifics of the IT-DOCS tool are discussed, including design constraints, implementation for various operating platforms, example script illustrations and output examples.

2. Some Common Goals of Scripting:

Scripts are created for many reasons. The common wisdom is “anything worth doing twice is worth scripting”.

Scripts are often created for **common operations**, any task that needs to be done frequently. Repetitive tasks and operations that have a high risk of error, either because of typo's, missed steps, or simply different people doing the same task correctly, but differently. Creation of new user accounts is a great example of this, where you may have many people, all creating accounts in slightly different ways as they navigate the standard operating system interfaces. This is generally an undesirable situation, especially in larger corporations, and the solution is often to script the task.

Rob VandenBrink. rvandenbrink@metafore.ca

Repetitive Tasks and Operations are another example where scripts can “come to the rescue”. A good example of this might be periodic audits of disk space used by user files, where a set of simple tasks (adding file space in a directory) must be executed hundreds or thousands of times to create a final report.

Repeatable operations are slightly different than repetitive operations. Repeatable operations are often complex tasks that must be repeated in exactly the same way in similar situations for later comparison against previous runs or an original benchmark. Scripts used in security or documentation audits, or audits of any kind, are an excellent example of this. An audit report is much more useful if the items being measured are measured in exactly the same way on successive runs. This allows successive audits to show either positive or negative progress. Also, if a complex script is written that measures compliance against a regulatory requirement, that same script can measure the same compliance metric at multiple clients as well as on successive runs for the same company.

Education is an important component that is often not discussed when describing scripts. The act of writing a script, especially a complex script that handles complex operations or situations, is a great way of learning aspects of an Operating System or environment that would otherwise never be explored. In addition, because scripts are usually collections of standard commands that can be run from the command line interface of the target operating system, a properly written, commented and documented script is an excellent education tool for other team-members, or for clients if the script is written as part of a consulting engagement.

Of course, these almost never occur in isolation. Many scripts replace repetitive, complex operations, that must be repeatable, and can also serve as an education conduit.

In all cases, the underlying intent of scripting an operation is to minimize the effort of completing the operation in the minimum amount of time, with no mistakes.

2.1. Documentation

A common use of scripting which is explored in this paper, is to create basic system documentation, which is a critical tool in maintaining a reliable system. This approach can be used to create documentation on a local host, or remotely over a network. Both approaches are discussed in this paper.

2.2. Audit for Compliance

Another common use of scripting, which is related to basic system documentation is to document a system with a final output expressing a compliance stance as compared to an external specification. Reports of this kind are often to compare a system's configuration against a regulatory compliance specification or security benchmark. This is also addressed in this paper, with a script measuring VMware ESX (version 3.5 or 4.0) compliance against the VMware Hardening Guide version 3.5, the current version as of the release of this paper.

2.3. Audit for Change Control

2.3.1. Zero day code

During a discussion about malware at a recent SANS class, the instructor (Ed Skoudis) remarked - "Sure you could catch zero days [malware] - just list all the processes on all the windows machines on your network, every day. You all know what's normal on your systems right?" (paraphrased) (Skoudis, 2008). This sparked an idea, which in turn became a SANS paper in it's own right.

In a recent SANS Technology Institute (<http://sans.edu>) paper, Jim McMillan and I wrote exactly this, a set of CMD files and WMIC scripts that inventory all running processes on all stations in a Windows domain. Known good processes are

Rob VandenBrink. rvandenbrink@metafore.ca

whitelisted in a text file, and known bad processes are similarly listed in a different text file. At each run, any new processes that have appeared since the last run are listed in a DIFF file.

The idea is that as the script is used over time, the whitelist file grows to include most (if not all) applications used in the corporation, and the daily DIFF report becomes very small.

This approach has worked extremely well. During initial testing (in a 100 user domain), the DIFF report between Days 1 and 2 identified 1 single net new process, which turned out to be the soft phone application used in the company. The Day 3 run found both an unapproved application and a windows host infected with malware.

The paper and script components can be found on the SANS.edu website, at http://www.sans.edu/resources/student_projects/200909_02.pdf.

In penetration tests, unauthenticated reconnaissance is almost always done early in the process. A script of this type is also included in IT-DOCS, a TCP connect scan is done of identified subnets. While maintaining a real-time representation of this is of questionable use, the intent of this is to integrate this scan into the change control loop described here. This then identifies new listening services on the network, in addition to net new running processes.

2.4. Scripting methods on various Operating Systems

Windows

Windows has a wide range of commands for displaying and modifying the configuration, including function specific commands such as “ipconfig”, more general command such as “net” or very wide scope command sets, with their own language, such as “netsh”. ADSI (Active Directory System Interface) offers both local and remote access to Active Directory functions and some host functions.

However, for this project, WMIC was chosen for most functions. WMIC (Windows management interface command) has access to the full WMI scope of functions, and permits export of local or remote windows configuration information to several different file formats, including flat text, CSV (Comma Separated Variables) and HTML.

Linux

In contrast to Windows, Linux does not have a general purpose API or command set that can be used for remote documentation. However, it does have several advantages in documentation. Almost everything in Linux is represented in a file. For instance, CPU utilization can be accessed by simply typing the file `/proc/cpuinfo`, and CPU information (speed, cache and model) can be displayed by typing the file `/proc/xasef`. Most configuration parameters for the Operating System are represented in a simple fashion in text files, located in `/etc`. In addition, Linux has a rich set of command line text manipulation facilities, featuring command such as `sed`, `awk`, `cut` and `tr`. Expert use of these tools allow pinpoint accuracy in extracting information from files in `/proc`, `/etc` or other locations, or output from any system command. Access to this full spectrum of tools is best accomplished using SSH. Full and mature support for SSH permits secure access to the system using any supported method.

Rob VandenBrink. rvandenbrink@metafore.ca

© 2010 SANS Institute, Author retains full rights.

VMware

The VMware ESX COS (Console Operating System) is very similar to Linux (specifically, Redhat Linux) in its documentation facilities, in that the command line has a rich set of Linux commands available to document the server. VMware specific commands are also available to document things such as Virtual Machine configuration, Virtual Switch configuration and connections, and other VMware specific functions.

However, the ESXi product is based on Busybox Linux rather than the Redhat base that the ESX COS enjoys. The ESXi product has a much more minimalistic and security conscious stance, to the point that direct remote access via SSH is not available on the product as shipped, and is not recommended to be enabled. This does not mean that there are no opportunities to script though. Scripting can be accomplished using the RCLI (Remote Command Line), using many of the same or similar commands as on ESXi. In addition, the RCLI function can be delivered within a virtual appliance called VMA (Vmware Management Appliance).

The RCLI and VMA methods are limited to accessing the ESX and ESXi servers in the farm. To access the higher level functions within vCenter, the best alternative remains Powershell. While Powershell is a powerful language, it is a large, object oriented language that poses a significant barrier to entry to many system administrators. In addition, all of the Powershell commands that access VMware functions are specific to the PowerCLI interface, so they are not familiar to most system administrators, in fact even the object oriented format and method of collecting data will be foreign to most. Finally, Powershell has a modular approach, where the PowerCLI is an imported module of commands. As more and more modules are in common use within powershell, we will find that powershell programs will or won't work

SSL and Telnet Enabled Devices and Hosts

Any device that can be administered remotely via telnet or SSH command line can easily be documented using that method. The client of choice in this situation would be plink.exe, the text-only terminal application that is part of the PUTTY suite of products (reference here), which supports both SSH and Telnet.

While Netware is reaching end of life, there is still in use by a large install base. Scripting in Netware is split into two sections. At the server command line, scripts are most often run at the local screen and keyboard, or scheduled via a native CRON facility.. There is an SSH function available, however, most Netware shops are in “don't rock the boat” mode. Their efforts are concentrated towards minimizing change until a migration project can be approved and started, rather than making non-critical changes to the existing environment.

2.5. Aspects of Scripting in Penetration Tests

There are two aspects to scripting in penetration testing. The more common aspect is the use of scripts in the penetration test itself. Scripting definitely addresses issues in penetration testing that we identified earlier in this paper – penetrations tests involve complex operations, which need to be both repeatable during the penetration test, and consistent between one penetration test and another. In addition, the scripts serve as a valuable education component, especially in consulting engagements when educating the client is always a positive thing to strive for.

For instance, the IT-DOCS toolkit outlined in this paper has been very valuable in penetration tests in several security engagements. A common timeline for an internal penetration test might be:

- Using a Windows host, get a DHCP IP Address
- Run “ipconfig /all”

- Using CAIN, run a Man in the Middle Attack on the first DNS Server in the workstation's list – this is almost always one of the Windows Domain Controllers.
- What is found in almost all cases is that System Administrators just can't help it – every problem or task seems to involve an RDP (Remote Desktop Protocol) session to their Domain Controller. Within an hour or so, you're almost certain to have the Domain Administrator password.
(<http://isc.sans.org/diary.html?storyid=7303>)
- Because administrators almost always use common passwords between platforms, you can now generally acquire most or all of the objectives of the Penetration Test, either directly or using other passwords that are now obtainable from the domain.
- With the formal pentest objectives (especially credentials) obtained, IT-DOCS has proven to be a great toolset to collect large volumes of data on the target environment in a very short period of time. By the end of Day 1, it's almost always possible to state that all the objectives are attained, and have several hundred pages of ancillary material to use as raw material for additional findings.

Of course, the use of IT-DOCS and similar tools has to fall within the bounds of the Statement of Work that describes the Penetration Test.

A much less common aspect of scripting and documentation in Penetration Testing is related to the fact that every IT department seems to have a subdirectory for IT information in the corporate share, often called "IS", "MIS", "IT" or similar. This directory almost always has detailed system documentation for almost every critical infrastructure component in the environment. In the case of IT-DOCS, this directory could be the output directory, which holds the documentation, or the input directory, which holds both the collection scripts and the credentials used to collect the configuration data. Also, if any scripts used for documentation happen to store temporary files on the hosts themselves (which IT-DOCS currently does in some cases), these are just as valuable. Finding such a subdirectory or file is a "finding" indeed.

Rob VandenBrink. rvandenbrink@metafore.ca

While in many engagements corporate documents “found” during an assessment cannot be opened, even the sight of a file or directory listing in a Penetration Test Report is enough to raise the blood pressure of any self-respecting IT Director.

3. The IT-DOCS tool

3.1. Implementation constraints

IT-DOCS was designed in the context of a consulting engagement. The tool had to be completed within a set period of time, in fact it had to be completed within the time estimated for a manual documentation process.

This meant that we had to use the methodology of “progressive elaboration” that is part of the Project Management Body of Knowledge (PMBOK) as discussed in SANS Management 525 course and GCPM certification. In the PMBOK context, progressive elaboration means that as a project progresses, the tasks at hand will have maximum detail associated with them, while tasks in the future will very likely not be fully defined until they are closer in the schedule.

Progressive elaboration in this project meant that existing reports, open source tools and operating system tools were all “in play” when designing version 1.0 of the application. As the application evolves, a common approach, common toolsets and especially a common output format will be defined and used across all target operation platforms. This approach means that version 1.0 of the tool, while not perfect, filled the immediate business requirement within the schedule and budget allocated.

3.2. Design

The IT-DOCS toolset has several basic design points:

Remote execution – It was decided early to, as much as possible, consolidate all code for documentation into a single repository. Running documentation scripts on the target hosts themselves not only complicates the build and deployment process for new hosts, it also makes it more likely to miss hosts in the documentation process. Centralizing all scripts and reports tends to make documentation a single process, rather than a process on each host.

No artefacts left on documented hosts – Leaving intermediate or final files associated with system documentation on target hosts is a clear security risk. A penetration tester or attacker browsing the filesystem who found such a file would then have critical information regarding the overall configuration of the target host, which might otherwise be much more difficult to collect.

Central collection of all data – From a customer perspective, it is highly desirable to browse documentation for all target systems from a single interface – in this case a simple 3-pane web page. It is critical, however, to implement proper authentication and filesystem security on this web site, ensuring that only authorized individuals have access to this sensitive information.

Common report format across all target platforms – The report formatting should be similar across all platforms. In Version 1 of IT-DOCS, the Linux configuration report and the VMware ESX Compliance report come very close to the final standard desired. Each section should have a title, a short description, the command used to collect the data, the audit finding (in the case of an audit report), and the raw data.

4. IT-DOCS Description - Version 1.0

4.1. Windows

The Windows documentation generated by version 1.0 of IT-DOCS is based mostly on WMI (Windows Management Interface), using the WMIC command. The current reports are all in WMI's native HTML format, which is at odds with all other IT-DOCS modules. A future version of IT-DOCS will use WMI's text output instead, and reformat it to match the other reports.

4.1.1. Windows Host Enumeration

The emphasis on version 1.0 of IT-DOCS was in documentation of servers, so a static list of target hosts is used for scheduled reporting in this version. However, in a future version of the tool, windows host enumeration could be done either by sweeping a set of subnets or by using Active Directory to identify domain members.

Sweeping a subnet could be done by using NMAP or some other port scanner, for instance by scanning for hosts with open "windows ports", such as 445/tcp.

However, as more and more windows hosts implement the Windows Firewall, this approach can become problematic. A more reliable method might be to use the DSQUERY command, which can be used to output the names of all hosts in the domain. The format of this command might be similar to this, which outputs the ip addresses of all computers in a domain (note that this command line is wrapped):

```
dsquery computer -s DCname -u domainname\administrator -p
adminpassword -limit 10000 | cut -d "," -f1 | cut -d "=" -f2
>>iplist.txt
```

this could be modified and run as "dsquery server ..." to output only server names in a domain.

However, this dsquery approach has the obvious problem of missing non-domain members entirely. The obvious solution is to implement both approaches, and use the best approach for any given situation.

4.1.2. Windows Host Assessment

As discussed, WMI is the mechanism chosen for Windows Host assessment. While the information of many WMI commands can be expressed by other command line utilities, WMI gives us a common method of collecting almost everything required, with a single syntax and a single output format. Most WMI inventory commands will look very similar. For instance, this one lists the BIOS version information of a target host (command line is wrapped):

```
wmic /output:%DIRNAME%\bios.htm /user:%UID% /password:%PWD%
/node:%IP% bios list brief /format:htable
```

where:

/output:%DIRNAME%\bios.htm	Command output is to be written to a file "bios.htm", in the directory specified by the %DIRNAME% variable
/user:%UID%	Login to the target host using the value of the %UID% variable as the username
/password:%PWD%	Use the %PWD% variable as the password
/node:%IP%	The target host is represented by the variable %IP%, but can be either the ip address, or a resolvable hostname (CN or FQDN).
bios list brief	This is the actual WMI command to execute – in this case, a brief listing of the bios information is requested.
/format:htable	The output format in this case is to an HTML table.

The WMI commands used for host documentation in Version 1.0 of IT-DOCS are:

qfe list brief	List all patches and Hotfixes
os get name, servicepackmajorversion	Get the Operating System and Service Pack version.
os list full	List Operating System Information
bios list brief	List BIOS version information
computersystem list	Basic system information
logicaldisk list brief	List logical disk partition information, including volume names, sizes and free space.
csproduct list brief	List installed products

Rob VandenBrink. rvandenbrink@metafore.ca

<code>group list</code>	List local groups. If the target is a domain controller, all groups in the domain are listed.
<code>useraccount list</code>	List local user accounts. If the target is a domain controller, all users in the domain are listed
<code>sysaccount list</code>	Built in system account status
<code>netclient list</code>	Network clients installed, which might include Microsoft networking client, NFS client, Terminal Services client, or in some cases Antivirus clients.
<code>share list</code>	List all shares on the host.
<code>nicconfig list</code>	List network card information
<code>product get vendor, name, version, installdate, packagecache, description, identifyingnumber</code>	List information on all installed applications.
<code>service list</code>	List all services, including current status and startup settings.

These commands can and should all be refined significantly – in many cases too much information is returned. For instance, “`wmic share list`” might be more succinctly stated “`wmic share get name, path`” and “`wmic nicconfig list`” might be more useful as “`wmic niconfig ipaddress,macaddress`”. In addition, some additional information might be required – for instance, more hardware information might be needed if IT-DOCS was to be used to evaluate which desktop computers might be due for replacement (by evaluating CPU hardware information or disk utilization statistics for instance).

In the next version of the tool, the scope of each tool will be evaluated in much more detail, and the format of the output will most likely be modified to comply with the final format standard.

4.1.3. Windows Active Directory Enumeration

As stated above, many Active Directory metrics can be reported on simply by taking host commands and directing them to a Domain Controller. User and Group lists for instance fall into this category.

Rob VandenBrink. rvandenbrink@metafore.ca

However, there are many Domain parameters that do not seem to be accessible from the WMIC command line tool. The DSQUERY command for instance can be useful in this regard. Some common DSQUERY commands (used by IT-DOCS) are listed here:

Dsquery computers	List all computers
Dsquery servers	List all servers
Dsquery ou	List all Organizational Units (OUs)

Group Policy is not nearly as well represented with scripting tools. In order to get supported scripting interface to Group Policy in windows 2003 for instance, you must install GPMC, which installs COM objects and a number of JSCRIPT and VBSCRIPT scripts that can in turn be used from the command line. However, in Server 2008, this function is moved over to powershell, where a series of cmdlets are available for Group Policy administration and reporting, after importing the group-policy module. This is described in detail at <http://technet.microsoft.com/en-us/library/ee461027.aspx> .

For instance, in Server 2003, the ListSOMPolicyTree script lists all links between AD Objects and Group Policy Objects. A partial listing is shown below

```
C:\Program Files\GPMC\Scripts>cscript /nologo ListSOMPolicyTree.wsf
=== GPO Links for domain THEDOMAIN ===
DC=THEDOMAIN
    GPO=Default Domain Policy
    GPO=Office2000
    GPO=Outlook-EditOptions
    GPO=IE-Options
    GPO=Outlook2002-SecuritySettings
    GPO=Citrix-UserConnectionSettings
    GPO=Citrix-NoIdleDisconnect
    GPO=Outlook2002-CheckSecuritySettingsFromExchange
OU=Domain Controllers
    GPO=Default Domain Controllers Policy
    GPO=Block-IE7-WindowsUpdate
    GPO=DNS-DynamicUpdateSingleNameDomain
OU=IRELAND USERS - NO INTERNET
    GPO=No Internet
    GPO=[Inaccessible]
OU=QA TESTERS
    GPO=QA TESTERS
OU=FLOOR USERS
    GPO=No Internet
    GPO=Floor Users - Least Restrictive
```

Rob VandenBrink. rvandenbrink@metafore.ca

```

    GPO=Floor Users
OU=Workstations
    GPO=Workstations
    GPO=Block-IE7-WindowsUpdate
    GPO=DST Registry Update and Refresh
    GPO=DNS-DynamicUpdateSingleNameDomain
OU=CitrixFarm
    GPO=Block-IE7-WindowsUpdate
    GPO=Citrix-DeleteCachedRoamingProfiles
    GPO=Citrix-DisableScreenSaver
    GPO=Citrix-FolderRedirection
    GPO=Citrix-HideSystemDrives
    GPO=Citrix-PreventAddPrinter
    GPO=Citrix-IE-BlockInternet
    GPO=Citrix-PreventControlPanel
    GPO=Citrix-IE-SecurityZones
    GPO=DISCONNECT Terminal connections

```

```

=== GPO Links for sites in forest DC=THEDOMAIN ===
    CN=CA
    CN=Default-First-Site-Name
    CN=MO
    CN=NC

```

```

C:\Program Files\GPMC\Scripts>cscript /nologo DumpGPOInfo.wsf
workstations

```

```

=====
Name:      Workstations
ID:       {0F58606C-F0ED-4968-9BF6-F188B46DE7D4}

```

```
-- Details --
```

```

Created:      3/1/2006 2:51:04 PM
Changed:     12/3/2009 12:13:16 PM
Owner:       THEDOMAIN\Domain Admins

```

```

User Enabled:  False
Mach Enabled:  True

```

```
-- Version Numbers --
```

```

User DS:      0
User Sysvol:  0
Mach DS:     7
Mach Sysvol:  7

```

```
-- Who this GPO applies to --
Authenticated Users

```

```
-- Who can edit this GPO --
```

```

-- Who can edit settings, modify security and delete this GPO --
Domain Admins
Enterprise Admins
SYSTEM

```

Rob VandenBrink. rvandenbrink@metafore.ca

```
-- Who only has Read access --  
ENTERPRISE DOMAIN CONTROLLERS
```

```
-- Who has custom permissions --
```

```
-- Where this GPO is linked (Sites,Domain,OU) --  
Workstations (OU)
```

```
=====
```

© 2010 SANS Institute, Author retains full rights.

The command "C:\Program Files\GPMC\Scripts> cscript GetReportsForAllGPOs.wsf c:\reports" outputs the actual settings in each Group Policy in the domain to both HTML and XML report formats – the output for a single policy is shown below:

Citrix-DeleteCachedRoamingProfiles			
Data collected on: 12/6/2009 12:35:39 PM			hide all
General			hide
Details			hide
Domain	THEDOMAIN		
Owner	THEDOMAIN\Domain Admins		
Created	9/20/2006 3:38:28 PM		
Modified	12/3/2009 12:13:16 PM		
User Revisions	0 (AD), 0 (sysvol)		
Computer Revisions	1 (AD), 1 (sysvol)		
Unique ID	{459C00EB-F0BC-45F3-B0A5-41D086C76FF7}		
GPO Status	Enabled		
Links			hide
Location	Enforced	Link Status	Path
CitrixFarm	No	Enabled	THEDOMAIN/CitrixFarm
CitrixServers	No	Enabled	THEDOMAIN/CitrixServers
This list only includes links in the domain of the GPO.			
Security Filtering			hide
The settings in this GPO can only apply to the following groups, users, and computers:			
Name			
NT AUTHORITY\Authenticated Users			
WMI Filtering			hide
WMI Filter Name	None		
Description	Not applicable		
Delegation			hide
These groups and users have the specified permission for this GPO			
Name	Allowed Permissions	Inherited	
THEDOMAIN\Domain Admins	Edit settings, delete, modify security	No	
THEDOMAIN\Enterprise Admins	Edit settings, delete, modify security	No	
NT AUTHORITY\Authenticated Users	Read (from Security Filtering)	No	
NT AUTHORITY\ENTERPRISE DOMAIN CONTROLLERS	Read	No	
NT AUTHORITY\SYSTEM	Edit settings, delete, modify security	No	
Computer Configuration (Enabled)			hide
Administrative Templates			hide
System/User Profiles			hide
Policy	Setting		
Delete cached copies of roaming profiles	Enabled		
User Configuration (Enabled)			hide
No settings defined.			

Rob VandenBrink. rvandenbrink@metafore.ca

4.1.4. Enumeration of Unauthorized Windows Software

In a recent paper, the discussion of using WMIC to enumerate a Windows domain for unauthorized software was discussed. The dsquery command is used to list all member computers, and with this list, each computer can then be queried using “wmic process list brief” or similar.

This process list is then evaluated in several ways. A blacklist file is maintained, listing known undesirable processes. A whitelist file is similarly created and maintained, listing all known good processes. At the beginning of the process, both files would be empty, but after the first run, a careful analysis of the output from one known good machine makes a good start on the whitelist file.

At this point, things can become very interesting. As daily runs accumulate, the “diff” command can be used to highlight new processes. These might be as a result of installing new software or patches, in which case they go into the whitelist file. However, in many cases these new processes will represent unauthorized software installed by the end user. In some cases these new processes will represent malware that has been missed by the antivirus software resident on the target computer.

When implemented on a typical domain, this approach can be used to locate unauthorized software or malware almost from Day 1.

This approach is not yet included in the IT-DOCS tool-set, but will certainly be added in the next version. All the code described in this approach is included in the paper that describes this set of scripts, located at

http://www.sans.edu/resources/student_projects/200909_02.pdf.

Rob VandenBrink. rvandenbrink@metafore.ca

4.2. Linux

For documentation of the Linux Operating System, a shell script called “sysinfo” authored by Makarand Hazarika was used

([http://developer.novell.com/wiki/index.php/Sysinfo - A Linux Box System Information Retriever](http://developer.novell.com/wiki/index.php/Sysinfo_-_A_Linux_Box_System_Information_Retriever), 2006). While this script is somewhat old, and originally intended for SUSE linux, it is easily adapted to almost any flavor of Linux. In addition, as a shell script it is easy to see all the commands and components involved in the final output. The script was used as downloaded from the Novell site, with the addition of a single command – “chkconfig -list”, which shows the startup status of all services on the system.

The approach of this script is very much in line with the overall goal of this project. The script is simple, the output shows the section title, the command used for the output, then the output itself. This not only documents the system, but adds to the education component of the entire project.

Since this script was selected late in the project, most of the other IT-DOCS reports do not share this format. The VMware ESX compliance report has a similar format, all other reports will be reformatted to this approach in version 2.0 of the toolset.

However, there are a few drawbacks to using this script. It runs as a shell script, and the output of the script is written to a directory on the target system. The intent of IT-DOCS is to run everything possible remotely, and to collect all output directly to a single repository without leaving any artefacts on the target systems. To this end, this script will be re-written using the plink single-line approach described in the SSH section.

4.3. VMWare ESX

4.3.1. Configuration Script

VMware ESX offers an extremely rich scripting interface, including the commands `vmware-cmd`, `vimsh`, the `esxconfig-*` commands, and especially the often overlooked `esxconfig-info` command. In addition, the `vm-support` script is the standard data collection script for VMware Technical Support, and can be used to collect a huge amount of information about the operating environment of the ESX server it is run on.

In addition to commands, simply browsing the files in `/etc`, especially the vmware-specific files in `/etc/vmware` such as `/etc/vmware/esx.conf` and `/etc/vmware/config` can yield a wealth of information. Listing the individual `.vmx` files for each virtual machine can also be useful.

Unfortunately, all of these “traditional” methods are in the process of being phased out. The COS (Console Operating System) of VMware ESX is being replaced by the much smaller ESXi COS. Configuration scripts for ESX will need to be writing using the RCLI (Remote Command Line) or VMA (VMware Management Appliance) interfaces. In addition, all vCenter components are native to the Windows Host that vCenter is installed on. The primary scripting interface for querying vCenter configurations remains Powershell.

The VMware ESX configuration script for IT-DOCS is currently in development, and will be a combination of RCLI or VMA, with a companion set of Powershell scripts.

4.3.2. Compliance Script – Compliance to VMware Hardening Guide

A script for auditing the security posture of an VMware ESX server was written as part of IT-DOCS. All sections within the hardening guide that are “script friendly” are covered. Each section is titled, and descriptions of each check and all commands

used are outlined in the output. When possible, a check for compliance is made, compliance status is red/green colour coded. Lastly, the raw data is shown.

Some sections of the VMware Hardening Guide are not "script friendly". For instance, the existence of a dedicated management network is not readily deduced from any script output. In cases such as this, the audit point is still addressed - the script outputs the raw information required for evaluation, with a description of the test's intent and goal. It is left to the auditor to make the final compliance decision based on that information.

4.4. SSL and Telnet Accessible Network Infrastructure

Data collection from any device that supports administrative access via SSL or Telnet is very simple. The PUTTY utility "PLINK" is used in both instances. For simple devices such as Routers, Switches, UPS PDU's, I/O redirection can be used. For instance, for a simple Router or Switch configuration information can be collected using a text file similar to rtrinven.in (below):

term len 0	Do not paginate the output. "term length 0" is the command on cisco routers, on HP or Brocade, you would use the "skip" command.
sho ver	Show the version information of the device
sho config	Show the configuration information of the device
exit	Logout of the device

And calling that text file using the syntax:

```
plink -l %1 -pw %2 %3 < rtrinven.in >%3_inventory.txt
```

However, as the complexity of the input file increases, it is common for plink to simply halt in midstream, and leaving the script in a state that never exits.

To solve this, there are two approaches. The simplest approach is to simply call PLINK for each single command, executing the required commands line-by-line. For instance, the command list above would be executed in a CMD file as:

```
plink -l %1 -pw %2 %3 "sho ver" >%3_inventory.txt
plink -l %1 -pw %2 %3 "sho config" >> %3_inventory.txt
```

Rob VandenBrink. rvandenbrink@metafore.ca

Where:

%1 is the first argument, the userid to login as

%2 is the password

%3 is the host identifier (ip address or resolvable host name)

Notice that when calling commands using this method, output pagination doesn't need to be considered.

A final approach is to encapsulate the entire ssh or telnet process within a higher level language, such as perl, expect or python. This approach runs counter to one of the major advantages of scripting however, namely education. Adding a higher level language simply makes it harder to understand and document the documentation process itself. In consulting, cross-training is often an essential part of the process – the client hires an expert so that not only does the job at hand get done well, but they have an opportunity to learn new methods and approaches along the way. Adding a higher level language simply makes that learning that much more difficult in many cases.

4.5. Netware

While Novell Netware is reaching it's end of life as far as Novell is concerned (March 2010 marks the date that Novell will stop supplying patches and security updates), it is still in use in some large environments who still need reliable, consistent documentation. The approach for documenting a Netware environment took a multi-pronged approach:

CONFIG.NLM, running on the Netware server console, is used to create a local config file. CONFIG.NLM is called using the command line arguments to collect information that is deemed important at that site. Common switches that might be considered for documentation purposes include:

Rob VandenBrink. rvandenbrink@metafore.ca

/s, to get all local server SET parameters.
 /e eDirectory information, and a list of partitions
 /f include all NCF files in C: and SYS: volumes
 /w include Java and Client executable files
 /j list all running Java processes
 /z include boot drive and netware storage device and volume information
 /m lists the versions of all running modules
 /ALL will list all this information, as well as pages and pages of other info.
 (<http://support.novell.com/docs/Readmes/InfoDocument/2974871.html>)

As can be seen, each additional configuration switch will add to the length of the report. Care should be taken that only the most useful information is collected, so that the report remains readable.

The CONFIG.NLM run is scheduled using CRON, first by ensuring that CRON is running (ensure that "LOAD CRON" is in the AUTOEXEC.NCF file), and by editing /etc/crontab, and adding the line:

```
1 0 * * * SYS:SYSTEM/CONFIG.NLM /sefm
```

which will run the CONFIG.NLM process at 1 minute past midnight, every day.

Netware commands are used to collect additional information. "CX" is used to collect NDS Tree information. NLIST is used to collect user information, including information specific to unused or disabled accounts, or accounts that do not require a password. NLIST is also used to collect group membership data. The RIGHTS command is used to collect rights assignments in the Netware filesystem. Specific command line switches for each command are listed in the Netware Scripts Appendix.

After all this is completed, the text outputs are converted to HTML to make viewing with a browser simpler.

5. IT-DOCS Development Plan

As noted in the implementation constraints, the toolset as it stands now is far from complete. The current script has different report formats across the different platforms,

5.1. Version 1.0

As discussed, version 1.0 of IT-DOCS was completed under a hard time limit. While all the target Operating Systems have documentation reports in the output, there are several issues with the current code:

- All of the credentials required to collect the documentation information from each OS are stored in a clear text file
- Linux reports are created on each local Linux host, and stored locally on a text file on each host for later collection.
- Windows reports are use the Microsoft HTML output format, which differs from all other reports.
- VMware reports are all done via shell scripts.
- Some popular operating environments are simply not represented. For instance, a request for AS/400 (IBM iSeries) support in the current version was deferred to a later version of the code. In addition, support has been requested for documentation of applications such as MS Exchange, MS SQL and other popular applications.

To this end, a roadmap for future development has been identified, outlined here. In all future releases, the report information itself will also be updated in each version going forward. For instance, many of the Windows reports simply contain too much information, or in some cases tables with empty columns. It's also very likely that there is information that is missed entirely. For instance, the current version does not contain any of the Active Directory Group Policy reports.

5.2. Version 2.0

There is an immediate need for changes to IT-DOCS before it can be used for other customer projects or environments.

Remote execution is required for the Linux reports. The function currently delivered by `sysinfo.sh` will be migrated to a remote `ssh` approach. This removes the script and more importantly the script output from the host being documented.

Reformat all windows reports to the common format. The current native Microsoft HTML does not match the standard IT-DOCS format. The windows reports should, for each audit point or section have a title, a short description of the section, the command used to create the output, and the output itself. This not only standardizes the output of the Windows reports, but more importantly meets the education requirement of the IT-DOCS package.

The VMware documentation and audit scripts should be re-written to run remotely. The current thinking is that `vCLI` (vSphere Command Line Interface) or `VMA` (vSphere Management Assistant) might be the desired development platform for these reports, instead of shell scripts. The stated direction of VMware is to move the ESX product towards the ESXi model (based on busybox linux), where shell scripts and `ssh` access are simply not viable options for documentation. `VMA` is actually a virtual appliance that offers an alternative to scripting, but keeps the approach of using common commands to create scripts.

Powershell, coupled with VMware's `PowerCLI` is also a very good alternative approach for a future direction. However powershell is simply too complex to satisfy the education component of the IT-DOCS tools. While Powershell is certainly available on all modern Windows platforms, complex Powershell applications are often not as portable as other scripting languages are, due to requirements for the implementation of exact versions of the appropriate Powershell bolt-ons make consistent delivery of a script to all clients problematic.

Rob VandenBrink. rvandenbrink@metafore.ca

While vCLI or VMA might be the preferred platforms for auditing or documenting individual ESX or ESXi hosts, these approaches cannot be used to document higher level vSphere constructs, such as DataCenter settings, Resource Pool settings, security within vCenter or settings on distributed virtual switches. For the higher level constructs within vSphere, Powershell is truly the only approach possible.

DIFF reports

The inclusion of DIFF reports in the documentation allows comparison of configuration settings between one day and another. This facilitates several things:

- Verification that approved changes were completed as described and on schedule
- Verification that unapproved changes are not occurring
- "Unapproved changes" certainly includes installation of unauthorized software, including malware in many cases.

5.3. Version 2.1

Encryption of all credentials stored on disk

As a security professional, storing all the user credentials for collection of the IT-DOCS data and reporting is an obvious problem. This approach was taken in the interest of getting a set of automated scripts working in as short a time as possible. However, cleartext credentials simply cannot be kept as a long or even short term solution. It's a obvious security problem, and would be a clear finding on any security audit based on any regulatory framework you'd care to name. All authentication and authorization information used by IT-DOCS should be encrypted on disk. The encryption method has not yet been selected.

Support for client certificates in the SSH collection routines will also be included at this time.

5.4. Version 3.0

Support for other Operating Systems

If interest is expressed, there is no reason that IT-DOCS can't be expanded to include a more diverse set of operating systems. Common server applications are also on the roadmap, and in fact some of these scripts are already started. For instance, Citrix XENAPP reports were completed for a different project, but are not yet in IT-DOCS tool. A list of future IT-DOCS targets might include:

Platforms:	Applications:
<ul style="list-style-type: none"> • OSX • IBM iSeries (AS/400) • Citrix XenApp / Presentation Server • XEN (XEN Server) • Citrix (XEN APP) • IBM pSeries (AIX) • Solaris • BSD Unix • Fiber Channel Switches (Brocade and Cisco) • SAN Support (IBM, HP, EMC) 	<ul style="list-style-type: none"> • MS Exchange • MS SQL Server • MS IIS • Domino • Oracle • MySQL

6. References

Skoudis, Ed. (2006). Windows Command-Line Kung Fu with WMIC . *SANS Internet Storm Center*, Retrieved from <http://isc.sans.org/diary.html?storyid=1229>

Skoudis, E. (2007). The Grammar of WMIC. *SANS Internet Storm Center*, Retrieved from <http://isc.sans.org/diary.html?storyid=2376>

Microsoft. (2009). Group Policy Management Console Scripting Samples. *Microsoft MSDN, Windows Developer Center*. Retrieved (2009, December 8) from <http://msdn.microsoft.com/en-us/library/aa814151%28VS.85%29.aspx>

Microsoft. (2009). Group Policy Cmdlets in Windows PowerShell. *Microsoft Technet, Windows Server TechCenter*. Retrieved (2009, December 8) from <http://technet.microsoft.com/en-us/library/ee461027.aspx>

Tatham, S. (2007). *Putty User Manual*. Retrieved from <http://the.earth.li/~sgtatham/putty/0.60/html/loc/>

Makarand, H, & Siddhartha, S. (2003, March 3). *Sysinfo - a Linux Box System Information Retriever*. Retrieved from <http://developer.novell.com/wiki/index.php/Sysinfo>

Skoudis, E. (2008, October). *Sans SEC560 - Network Penetration Testing and Ethical Hacking*.

McMillan, J, & VandenBrink, R. (2009). *STI Student Group Written Project - Downadup / Conficker Incident Report*. Retrieved from http://www.sans.edu/resources/student_projects/200909_02.pdf

Rob VandenBrink. rvandenbrink@metafore.ca

Warlick, David (2004). Son of Citation Machine. Retrieved February 17, 2009, from Son of citation machine Web site: <http://www.citationmachine.net>

© 2010 SANS Institute, Author retains full rights.

1. Appendix: VMware ESX Compliance Script and Output – Evaluating Compliance to VMware Hardening Guide version 3.5

This shell script outputs HTML, it is normally called as `./esxaudit.sh >auditoutput.html` Where “auditoutput.html” is collected after execution to a central repository, generally using a CRON job that calls scp. The output of this job on a test ESX host follows immediately after the commented script.

<code>#!/bin/bash</code>	
<code>echo "<pre>"</code>	<code><pre></code> is an html keyword that indicates the output is pre-formatted, so that end of line characters are handled with carriage-return/linefeed instead of html markup <code><p></code> and <code></p></code>
<code>echo -n "<H1> Compliance Report for ESX Host - "</code> <code>hostname</code>	No newline on this, so that the hostname is appended to the line
<code>echo </H1></code>	
<code>echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Inventory ESX Version and Patch Levels</p></H3>"</code>	Describe the first test (note the colour changes throughout)
<code>echo "Test Commands cat /etc/vmware-release"</code>	Outline the commands used
<code>echo " esxupdate -l query"</code>	
<code>echo "Test Results"</code>	Show the results
<code>echo</code>	
<code>echo</code>	
<code>cat /etc/vmware-release</code>	
<code>echo "<HR>"</code>	
<code>esxupdate -l query</code>	
<code>echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Inventory Guests on this ESX Host</p></H3>"</code>	
<code>echo "Test Command: Vmware-cmd -l"</code>	
<code>echo "Test Results: "</code>	
<code>echo "<HR>"</code>	
<code>for vm in `vmware-cmd -l`</code>	"vmware-cmd -l" lists all guests
<code>do</code>	
<code>name=`vmware-cmd "\$vm" getconfig displayname -q`</code>	For each guest:
<code>echo VM Displayname \$name</code>	Echo the displayname
<code>echo VM Path \$vm</code>	Echo the path
<code>echo -e \\n</code>	
<code>done</code>	
<code>echo</code>	
<code>echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Inventory Virtual Switches on each ESX Host</p></H3>"</code>	Switch configurations
<code>echo "Test Commands esxcfg-vswitch -l"</code>	
<code>echo "<HR>"</code>	
<code>esxcfg-vswitch -l</code>	List all vSwitches
<code>echo</code>	
<code>echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Inventory Guest VM Virtual Switch Connectivity</p></H3>"</code>	List the vSwitch connectivity of each guest

echo "Test Commands: cat /pathspec/guestname.vmx grep networkName"	
echo "Test Results:"	
echo "<HR>"	
for vm in `vmware-cmd -l`	
do	
name=`vmware-cmd "\$vm" getconfig displayname -q`	
echo \$name	
echo -e `cat \$vm grep networkName`	vSwitch connectivity is done by simply dumping the vmx file and grepping out the connectivity. This is the simplest way I've found (so far) to handle absolutely every case, including hosts with multiple links.
echo	
done	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify Guest VM Cut / Paste Restrictions</p></H3>"	This is the first test where we test for actual compliance.
echo " Test Detailed Description: Disable copy and paste operations for the guest operating system between the guest OS and the Remote Console"	
echo " Commands to Test (for each VM):"	
echo "cat vmname.vmx grep isolation.tools.copy"	
echo "cat vmname.vmx grep isolation.tools.paste "	
echo "cat vmname.vmx grep isolation.tools.setGUI"	
echo "Blank Output indicates default settings (cut/paste permitted)"	
echo "<HR>"	
for vm in `vmware-cmd -l` ;	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
val1=`cat \$vm grep isolation.tools.copy grep -iv true wc -l`	
val2=`cat \$vm grep isolation.tools.paste grep -iv true wc -l`	
val3=`cat \$vm grep isolation.tools.setGUI grep -iv true wc -l`	
iscomp=\$((\$val1+\$val2+\$val3))	
if [\$iscomp -lt 3] ; then	Here is the test
echo "NONCOMPLIANT"	Noncompliant status in red
echo "Non Compliant settings:"	
if [\$val1 -eq 0] ; then echo "\"isolation.tools.copy=true\" is required" ;	
fi	
if [\$val2 -eq 0] ; then echo "\"isolation.tools.paste=true\" is required"	
; fi	
if [\$val3 -eq 0] ; then echo "\"isolation.tools.setGUI=true\" is required" ; fi	
else	
echo "COMPLIANT"	Compliant status in green
fi	
done	
echo	
echo	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Are Guest VMs Prevented from writing name/value pairs to config file?</p></H3>"	
echo "Test Commands - for each VM"	
echo "cat \$vm grep tools.setinfo.disable"	
echo "Validation Criteria: these should be set to true. Null returns indicate the default (false)"	
echo "<HR>"	

for vm in `vmware-cmd -l`	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
val1=`cat \$vm grep tools.setinfo.disable grep -i true wc -l`	
if [\$val1 -eq 0] ; then	
echo "NONCOMPLIANT"	
echo "\"tools.setinfo.disable=true\" is required"	
echo "Noncompliant settings:"	
cat \$vm grep tools.setinfo.disable	
else	
echo "COMPLIANT"	
fi	
done	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Guest log Size and Rotation</p></H3>"	This test is against VMware's numbers – different organizations may elect to use different values. This script is against VMware's documentation.
echo "Test Detail"	
echo "Logs should be rotated when they reach a specific size (100k), rotated to keep old logs (10)"	
echo "These are the values recommended in the VMware hardening guide, some organizations may elect to set different values as standards."	
echo "Validation Criteria:"	
echo "log.rotateSize = 100000"	
echo "log.keepOld = 10"	
echo "tools.setinfo.sizeLimit = 1048576"	
echo "<HR>"	
for vm in `vmware-cmd -l`	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
cat \$vm grep log.rotate	
cat \$vm grep log.keep	
cat \$vm grep sizeLimit	
done	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify Guest Non-Persistent Disks</p></H3>"	
echo "Test Command: cat guestvmxfile.vmx grep scsi grep .mode"	
echo "Verify values:"	
echo "Default Value (blank) indicates independant-nonpersistent mode"	
echo "Blank or \"independant-nonpersistent\" indicates compliance."	
echo "<HR>"	
for vm in `vmware-cmd -l`	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
val1=`cat \$vm grep scsi grep .mode grep -iv independant-persisent wc -l`	
if [\$val1 -eq 0] ; then	
echo "COMPLIANT"	
else	
echo "NONCOMPLIANT"	
fi	
cat \$vm grep scsi grep .mode	
done	

<pre> echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Guest Unauthorized device connection</p></H3>" echo "Test Detail: Ensure that unauthorized devices are not currently connected to guest VMs" echo "Test Command: cat guestvmxfile.vmx grep .present" echo "Verify values: Varies by organization" echo "<HR>" </pre>	
<pre> for vm in `vmware-cmd -l` do echo `vmware-cmd "\$vm" getconfig displayname -q` cat \$vm grep .present done </pre>	
<pre> echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Guest Prevent Unauthorized Removal or Connection of Devices</p></H3>" echo "Test Command: For each VM: cat <vmname.vmx> grep isolation.tools.disable " echo "Verify Values: should be set to \"true\". \"false\" or blank indicates non-compliance." echo "<HR>" for vm in `vmware-cmd -l` do echo `vmware-cmd "\$vm" getconfig displayname -q` val1=`cat \$vm grep isolation.tools.disable grep -iv true wc -l` if [\$val1 -eq 0] ; then echo "NONCOMPLIANT" echo "Noncompliant settings (blank line indicates noncompliance):" cat \$vm grep isolation.tools.disable else echo "COMPLIANT" fi done </pre>	
<pre> echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Guest Denied Disk Operations D.O.S.</p></H3>" echo "Test Command: cat \$vm grep isolation.tools.disk" echo "Verify Values:: should be set to \"true\". \"false\" or blank indicates non-compliance." echo "<HR>" for vm in `vmware-cmd -l` do echo `vmware-cmd "\$vm" getconfig displayname -q` cat \$vm grep isolation.tools.disk done </pre>	
<pre> echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify that Guest OS Matches ESX Configuration</p></H3>" echo "Test Command: cat <guestname>.vmx grep guestOS sort -r" echo "Verify Values:: should be the same value as the actual OS of the Guest VM." echo "<HR>" for vm in `vmware-cmd -l` do echo `vmware-cmd "\$vm" getconfig displayname -q` cat \$vm grep guestOS sort -r </pre>	

done	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Guest File Permissions</p></H3>"	
echo "Test Command: ls -lh <questname>.vmx"	
echo "Verify Values:: user and group should be root and root"	
echo " permissions should be -rwxr-xr-x (755)"	
echo "<HR>"	
for vm in `vmware-cmd -l`	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
ls -lh \$vm	
done	
echo " list VMDK file permissions"	
echo "Verify Values:: user and group should be root and root"	
echo " permissions should be -rw-----"	
echo "<HR>"	
for vm in `vmware-cmd -l`	
do	
echo `vmware-cmd "\$vm" getconfig displayname -q`	
for vmdk in `cat \$vm grep vmdk`	
do	
ls -lh \$vmdk	
done	
done	
echo "<H2>SERVICE CONSOLE checks</H2>"	
echo	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Firewall Checks</p></H3>"	This one is another "auditor makes the call" test. The raw data is printed, the auditor must interpret.
echo "Test Command: esxcfg-firewall -q"	
echo "Verify Values: Firewall settings should meet corporate standards"	
echo "<HR>"	
esxcfg-firewall -q	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Running Service Checks</p></H3>"	This is another one that is hard to test for. Services for server hardware management make it difficult to arrive at a list of "known benign" services. This list will vary between corporations, but more especially between server hardware vendors.
echo "Test Command: chkconfig --list"	
echo "Verify Values:: Running Services should meet corporate standards"	Verify services and run state- again, verify against corporate standards
echo "<HR>"	
chkconfig --list	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: List Running Processes</p></H3>"	
echo "Test Command: ps -efw --forest"	
echo "Verify Values:: Check process list for unexpected	

processes"	
echo "<HR>"	
ps -efw --forest	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Control of SSH Root Privileges</p></H3>"	Verify root privileges. The default settings for this are correct in ESX, but this is a good lab test and is in the Hardening Guide.
echo "Test Command: Cat /etc/ssh/sshd_config grep Root "	
echo "Verify Values:: PermitRootLogin should be set to no "	
echo "<HR>"	
cat /etc/ssh/sshd_config grep Root	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify that ftp is disabled by port</p></H3>"	Is FTP disabled?
echo "Test Command: netstat -na grep LIST grep \:21"	
echo "Verify Values:: output should be blank"	
echo "<HR>"	
netstat -na grep LIST grep \:21	
echo	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description</p></H3>: Verify that ftp is disabled by configuration"	
echo "Test Command: ls -l /etc/xinetd.d grep ftp"	
echo "Verify Values:: output should be blank"	
echo "<HR>"	
ls -l /etc/xinetd.d grep ftp	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description</p></H3>: verify that root cannot login directly to the console"	Can root log into the console directly?
echo "Test Command: cat /etc/security"	
echo "Verify Values:: this listing should be empty"	
echo "<HR>"	
echo "Test Results:"	
val1=`cat /etc/security wc`	
if [val1 -gt 0] ; then	
echo "NONCOMPLIANT"	
else	
echo "COMPLIANT"	
fi	
echo "Raw Data:"	
cat /etc/security	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description</p></H3>: verify su permissions are restricted"	
echo " enumerate members of the wheel group"	
echo "Test Command: cat /etc/group grep wheel"	
echo "Verify Value: root only, or authorized users only"	
echo "<HR>"	
cat /etc/group grep wheel	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify that only the wheel group members have access to su</p></H3>"	Who has access to the "su" command?
echo "Test Command: cat /etc/pam.d/su grep auth grep	

pam_wheel.so grep -v \#"	
echo "<HR>"	
echo "Test Results:"	
val1=`cat /etc/pam.d/su grep auth grep pam_wheel.so grep -v \# wc -l`	
if [\$val1 -gt 0] ; then	
echo "NONCOMPLIANT"	
else	
echo "COMPLIANT"	
fi	
echo "Raw Data:"	
cat /etc/pam.d/su grep auth grep pam_wheel.so grep -v \#	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description</p></H3>: verify that directory service is used for sudo authentication"	Is a directory service used for sudo authentication?
echo "Test Command: cat /etc/pam.d/sudo"	
echo "Verify Value: should read \"service=system-auth\""	
val1=`cat /etc/pam.d/sudo grep system-auth wc -l`	
if [val1 -gt 0] ; then	
echo "NONCOMPLIANT"	
else	
echo "COMPLIANT"	
fi	
echo "Raw Data:"	
cat /etc/pam.d/sudo grep system-auth	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: verify that password aging is enabled</p></H3>"	Is password aging enabled?
echo "Test Command: cat /etc/passwd cut -d : -f 1 (get userids)"	
echo " chage -l <userid> (for each userid, check age)"	
echo "<HR>"	
for id in `cat /etc/passwd cut -d : -f 1`	
do	
echo =====	
echo "Userid = \$id"	
chage -l \$id	
done	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify Account Lockout</p></H3>"	Is account lockout configured?
echo "Test Command: cat /etc/pam.d/system-auth grep pam_tally"	
echo "Verify Values::"	
echo "for auth required: /lib/security/pam_tally.so no_magic_root"	
echo "for account required: /lib/security/pam_tally.so deny=3 no_magic_root"	
echo "No returned value indicates non-compliance"	
echo	
val1=`cat /etc/pam.d/system-auth grep pam_tally.so wc -l`	

if [\$val1 -gt 0] ; then	
echo "COMPLIANT"	
else	
echo "NONCOMPLIANT"	
fi	
echo "Raw Data:"	
cat /etc/pam.d/system-auth grep pam_tally.so	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify logging of failed logins</p></H3>"	Are failed logins logged?
echo "Test Command: ls /var/log/faillog wc -l"	
echo "Verify Values:: 1 file exists, compliant"	
echo " 0 file does not exist, noncompliant"	
val=`ls /var/log/faillog wc -l`	
if [\$val1 -gt 0] ; then	
echo "NONCOMPLIANT"	
else	
echo "COMPLIANT"	
fi	
echo "<H2>MAINTAINING PROPER LOGGING</H2>"	Is NTP running?
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify that NTP is running?</p></H3>"	
echo "Test Command: ps -efw grep ntpd wc -l"	
echo "Verify Values:: 1 = ntpd is running, compliant"	
echo " 0 = ntpd is not running, noncompliant"	
echo	
val1=`ps -efw grep ntpd wc -l`	
if [\$val1 -gt 0] ; then	
echo "COMPLIANT"	
else	
echo "NONCOMPLIANT"	
fi	
echo "<H3><HR><p style='background-color: #C0C0C0;'>Test Description: Verify that NTP is associated with a trusted timesource?</p></H3>"	Is NTP associated with a trusted timesource? (manual verify)
echo "Test Command: ntpq associations"	
echo "Verify Value: trusted timesource(s) configured"	
echo " active ntp associations"	
echo " firewall permissions show ntpClient"	
echo ntp.conf server settings:	
cat /etc/ntp.conf grep -i server	
echo "ntp/step-tickers server settings:"	
cat /etc/ntp/step-tickers	
echo	
echo Service Startup Settings	
chkconfig --list grep ntp	
echo	
echo Firewall Settings	
esxcfg-firewall grep ntpClient	
echo	
echo "NTP Associations:"	
ntpq -p	
echo	

echo "<H3><HR><p style='\"background-color: #C0C0C0;\">Test Description: are local logs being written?</p></H3>"	
echo "Test Command: date ; echo ; ls -lhst /var/log head -25"	
echo "Verify Values:: Compare file date against current date"	
echo " compare file dates against current date"	
echo -n "Current Date is: " ; date ; echo ; ls -lhst /var/log head -25"	
echo "<H3><HR><p style='\"background-color: #C0C0C0;\">Test Description:</p></H3> Verify that log file growth is controlled"	Is log file growth controlled? This one has some serious fun with grep!
echo "Test Command: cat /etc/logrotate.conf"	
echo "Test Results:"	
echo "<HR>"	
for fn in vmkernel vmksummary vmkwarning	
do	
echo -n "Size limit: " ; cat /etc/logrotate.d/\$fn grep size grep -v \# grep -v compress	
echo -n "Filename \$fn Compression Check: "	
cat /etc/logrotate.d/\$fn grep size grep -v \# grep compress grep -v nocompres	
cat /etc/logrotate.d/\$fn	
if [`cat /etc/logrotate.d/\$fn grep size grep -v \# grep compress grep - v nocompres wc -l` -lt 1] ; then	
echo Noncompliant	
else	
echo Compliant	
fi	
done	
echo "<H3><HR><p style='\"background-color: #C0C0C0;\">Test Description: is remote syslog running?</p></H3>"	Is remote syslog configured?
echo "Test Command: cat/etc/syslog.conf grep @"	
echo "Test Verify: view syslog.conf, verify remote logging \(@ symbol\)"	
echo "Test Results:"	
echo "<HR>"	
if [`cat /etc/syslog.conf grep @ wc -l` -gt 0] ; then	
echo "COMPLIANT"	
else	
echo "NONCOMPLIANT"	
fi	
echo "Test Raw Output:"	
cat /etc/syslog.conf grep @	
echo "<H3><HR><p style='\"background-color: #C0C0C0;\">Test Description: Verify correct permissions on VMware utility file permissions</p></H3>"	Do the vmware utility files have the correct permissions?
echo "Test Command: ls -lh /usr/sbin/esxcfg-*"	
echo "Verify Value: all /usr/sbin/esxcfg-* commands are set to -r- x----- (500)"	
echo " except esxcfg-auth which should be -r-xr--r-- (544)"	
echo -n "Test Results: "	
echo "<HR>"	
echo -n "esxcfg* permissions - "	
if [`ls -lh /usr/sbin/esxcfg-* grep -v esxcfg-auth grep -v \'-r-x-l-x-l-l-l-l-l-l\'	

<pre>wc -l -gt 0] ; then echo "NONCOMPLIANT" else echo "COMPLIANT" fi</pre>	
<pre>echo -n "Test Results: " echo -n "esxcfg-auth permissions - " if [`ls -lh /usr/sbin/esxcfg-auth grep -v "\-r\-\-r\-\-r\-\-r\-\-" wc -l` -gt 0] ; then echo "NONCOMPLIANT" else echo "COMPLIANT" fi</pre>	
<pre>echo echo "Test Raw Data: " ls -lh /usr/sbin/esxcfg-*</pre>	
<pre>echo "<H3><HR><p style='background-color: #C0C0C0;\`>Test Description: Verify correct file permissions on logs</p></H3>" echo "Test Commands ls -lh /var/log" echo " ls -lh /var/log/vmware" echo " ls -lh /var/log" echo " ls -Rlh /vmfs/volumes/" echo "Verify Values:: all log files have permissions of -rw----- (600)" echo " except: /var/log/vmware/webAccess/* -rw-r--r-- (755)" echo " and: Individual Virtual Machine log files -rw-r--r-- (644)" echo</pre>	Do log files have the correct permissions?
<pre>echo echo echo -n "Test Results: " echo "<HR>" echo "in /var/log:" if [`ls -lh /var/log grep -v drw grep -v txt grep -v html grep -v wtmp grep -v rpmpkgs grep -v dme grep -v lastlog grep -v "\-rwl\-\-\-\-\-\-" grep -v total wc -l` -gt 0] ; then echo "NONCOMPLIANT" echo echo Non Compliant Files: ls -lh /var/log grep -v drw grep -v txt grep -v html grep -v wtmp grep -v rpmpkgs grep -v dme grep -v lastlog grep -v "\-rwl\-\-\-\-\-\-" grep -v total echo echo "COMPLIANT" fi echo</pre>	Lots more grep !!
<pre>echo "Test Raw Data:" echo ls -lh /var/log grep -v drw grep -v txt grep -v html grep -v wtmp grep -v rpmpkgs grep -v dme grep -v lastlog grep -v total</pre>	
<pre>echo "In /var/log/vmware/webAccess:" if [`ls -lh /var/log/vmware/webAccess grep -v "\-rwl\-\-\-\-\-\-" wc -l` -gt</pre>	This might be better in octal, but is more understandable in ascii

0] ; then	
echo "NONCOMPLIANT"	
echo	
echo Noncompliant files:	
ls -lh /var/log/vmware/webAccess grep -v "\-rw\-\-\-\-\-\-	
else	
echo "COMPLIANT"	
fi	
echo	
echo "Test Raw Data:"	
ls -lh /var/log/vmware/webAccess	
echo	
echo	
echo "Individual VM Logs"	
if [`ls -Rlh /vmfs/volumes/ grep \.log grep -v "\-rw\-\-\-\-\-\-	
] ; then	
echo "COMPLIANT"	
else	
echo "NONCOMPLIANT"	
echo "Noncompliant files:"	
ls -Rlh /vmfs/volumes/ grep \.log grep -v "\-rw\-\-\-\-\-\-	
fi	
echo "Raw Data:"	
ls -Rlh /vmfs/volumes/ grep \.log	
echo "<H3><HR><p style='background-color: #C0C0C0;'\>Test Description: Is root filesystem protected from filling up?</p></H3>"	Is the filesystem partitioned to protect the root from filling up?
echo "verify that /home, /tmp and /var/log are in 3 separate, non-root partitions"	
echo "Test Command: vdf -lh"	
echo "Verification Value: verify a separate entry for each critical partition: /var/log, /home, tmp"	
echo "Test Results"	
echo "<HR>"	
echo -n "/var/log partition is "	
if [`vdf -h grep /var/log wc -l ` -gt 0] ; then	
echo "COMPLIANT "	
else	
echo "NONCOMPLIANT "	
fi	
echo -n "/home partition is "	
if [`vdf -h grep /home wc -l ` -gt 0] ; then	
echo "COMPLIANT "	
else	
echo "NONCOMPLIANT "	
fi	
echo -n "/tmp partition is "	
if [`vdf -h grep /tmp wc -l ` -gt 0] ; then	
echo "COMPLIANT "	
else	
echo "NONCOMPLIANT "	
fi	
echo "Test Raw Data:"	
vdf -h	

1. Compliance Report for ESX Host -

1.1.1.

Test Description: Inventory ESX Version and Patch Levels

Test Commands cat /etc/vmware-release
esxupdate -l query

Test Results

VMware ESX Server 3

Installed software bundles:

----- Name -----	---	Install	Date ---	--- Summary ---
3.5.0-64607		19:34:53	10/10/08	Full bundle of ESX 3.5.0-64607
ESX350-200802303-SG		19:34:54	10/10/08	util-linux security update
ESX350-200802304-SG		19:34:54	10/10/08	perl security update
ESX350-200802305-SG		19:34:55	10/10/08	openssl security update
ESX350-200802306-BG		19:34:55	10/10/08	tzdata update
ESX350-200802408-SG		19:34:55	10/10/08	Security Updates to the Python
Package.				
ESX350-200803209-UG		19:34:55	10/10/08	Update to the ESX Server Service
Console				
ESX350-200803210-UG		19:34:56	10/10/08	Update to VMware-esx-drivers-net-bnx2
ESX350-200803212-UG		19:34:56	10/10/08	Update VMware qla4010/qla4022 drivers
ESX350-200803213-UG		19:34:56	10/10/08	Driver Versioning Method Changes
ESX350-200803214-UG		19:34:57	10/10/08	Update to Third Party Code Libraries
ESX350-200804404-BG		19:34:57	10/10/08	Update to VMware-esx-drivers-scsi-
vmkisc				
ESX350-200804405-BG		19:34:57	10/10/08	Update to VMware-esx-drivers-scsi-
megara				
ESX350-200805504-SG		19:34:57	10/10/08	Security Update to Cyrus SASL
ESX350-200805505-SG		19:34:58	10/10/08	Security Update to unzip
ESX350-200805506-SG		19:34:58	10/10/08	Security Update to Tcl/Tk
ESX350-200805507-SG		19:34:58	10/10/08	Security Update to krb5
ESX350-200805514-BG		19:34:59	10/10/08	Update to VMware-esx-drivers-net-e1000
ESX350-200808201-UG		19:34:59	10/10/08	Updates VMkernel, Service Console,
hostd				
ESX350-200808202-UG		19:34:59	10/10/08	Update to ESX Scripts
ESX350-200808203-UG		19:34:59	10/10/08	Update to Backup Tools
ESX350-200808205-UG		19:35:00	10/10/08	Updates to CIM and Pegasus
ESX350-200808206-UG		19:35:00	10/10/08	Update to vmware-hwdata
ESX350-200808207-UG		19:35:00	10/10/08	Update to VMware-esx-lnxcfg
ESX350-200808209-UG		19:35:01	10/10/08	Update to VMware-esx-srvrmgmt
ESX350-200808210-UG		19:35:01	10/10/08	Update to VMware-esx-drivers-net-ixgbe
ESX350-200808211-UG		19:35:01	10/10/08	Update to the tg3 Driver

Rob VandenBrink. rvandenbrink@metafore.ca

```

ESX350-200808212-UG    19:35:02 10/10/08 Update to the MegaRAID SAS Driver
ESX350-200808213-UG    19:35:02 10/10/08 Update to the MPT SCSI Driver
ESX350-200808214-UG    19:35:02 10/10/08 Update to the QLogic SCSI Driver
ESX350-200808215-UG    19:35:02 10/10/08 Update to the Emulex SCSI Driver
ESX350-200808217-UG    19:35:03 10/10/08 Update to Web Access
ESX350-200808218-UG    19:35:03 10/10/08 Security Update to Samba
ESX350-Update-02      19:35:03 10/10/08 ESX Server 3.5.0 Update 2

```

New packages:

```
VMware-vpxa-2.5.0-119598
```

1.1.2.

Test Description: Inventory Guests on this ESX Host

Test Command: `Vmware-cmd -l`

Test Results:

VM Displayname VM04

VM Path `/vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/VM04/VM04.vmx`

VM Displayname VM03

VM Path `/vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/VM03/VM03.vmx`

VM Displayname SRV01

VM Path `/vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/SRV01/SRV01.vmx`

VM Displayname FW01

VM Path `/vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/FW01/FW01.vmx`

1.1.3.

Test Description: Inventory Virtual Switches on each ESX Host

Test Commands `esxcfg-vswitch -l`

Switch Name	Num Ports	Used Ports	Configured Ports	MTU	Uplinks
vSwitch0	64	4	64	1500	vmnic0

PortGroup Name	VLAN ID	Used Ports	Uplinks
Virtual Machine Network0		0	vmnic0
Service Console	0	1	vmnic0

Rob VandenBrink. rvandenbrink@metafore.ca

Switch Name	Num Ports	Used Ports	Configured Ports	MTU	Uplinks
vSwitch1	64	1	64	1500	

PortGroup Name	VLAN ID	Used Ports	Uplinks
Internal Trust Zone	0	0	

Switch Name	Num Ports	Used Ports	Configured Ports	MTU	Uplinks
VSwitch3	64	3	64	1500	vmnic1

PortGroup Name	VLAN ID	Used Ports	Uplinks
Public Internet Trust Zone0		0	vmnic1

Switch Name	Num Ports	Used Ports	Configured Ports	MTU	Uplinks
vSwitch3	64	1	64	1500	

PortGroup Name	VLAN ID	Used Ports	Uplinks
DMZ Trust Zone	0	0	

Switch Name	Num Ports	Used Ports	Configured Ports	MTU	Uplinks
vSwitch2	64	4	64	1500	vmnic2

PortGroup Name	VLAN ID	Used Ports	Uplinks
VMkernel	0	1	vmnic2

1.1.4.

Test Description: Inventory Guest VM Virtual Switch Connectivity

Test Commands: `cat /pathspec/guestname.vmx | grep networkName`

Test Results:

VM04
 ethernet0.networkName = "Public Internet Trust Zone"

VM03
 ethernet0.networkName = "Internal Trust Zone"

SRV01
 ethernet0.networkName = "DMZ Trust Zone"

FW01
 ethernet0.networkName = "Virtual Machine Network" ethernet1.networkName =
 "Virtual Machine Network" ethernet2.networkName = "Virtual Machine Network"

1.1.5.

Test Description: Verify Guest VM Cut / Paste Restrictions

Test Detailed Description: Disable copy and paste operations for the guest operating system between the guest OS and the Remote Console

Commands to Test (for each VM):

```
cat vmname.vmx | grep isolation\.tools\.copy
cat vmname.vmx | grep isolation\.tools\.paste
cat vmname.vmx | grep isolation\.tools\.setGUI
Blank Output indicates default settings (cut/paste permitted)
```

VM04

NONCOMPLIANT

Non Compliant settings:

```
"isolation.tools.copy=true" is required
"isolation.tools.paste=true" is required
"isolation.tools.setGUI=true" is required
```

VM03

NONCOMPLIANT

Non Compliant settings:

```
"isolation.tools.copy=true" is required
"isolation.tools.paste=true" is required
"isolation.tools.setGUI=true" is required
```

SRV01

NONCOMPLIANT

Non Compliant settings:

```
"isolation.tools.copy=true" is required
"isolation.tools.paste=true" is required
"isolation.tools.setGUI=true" is required
```

FW01

NONCOMPLIANT

Non Compliant settings:

```
"isolation.tools.copy=true" is required
"isolation.tools.paste=true" is required
"isolation.tools.setGUI=true" is required
```

1.1.6.

Test Description: Are Guest VMs Prevented from writing name/value pairs to config file?**Test Commands - for each VM**

```
cat /vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/FW01/FW01.vmx | grep
tools.setinfo.disable
```

Validation Criteria: these should be set to true. Null returns indicate the default (false)

VM04

NONCOMPLIANT

"tools.setinfo.disable=true" is required

Noncompliant settings:

VM03

NONCOMPLIANT

"tools.setinfo.disable=true" is required

Noncompliant settings:

SRV01

NONCOMPLIANT

"tools.setinfo.disable=true" is required

Noncompliant settings:

FW01

NONCOMPLIANT

"tools.setinfo.disable=true" is required

Noncompliant settings:

1.1.7.

Test Description: Guest log Size and Rotation

Test Detail

Logs should be rotated when they reach a specific size (100k), rotated to keep old logs (10)

These are the values recommended in the VMware hardening guide, some organizations may elect to set different values as standards.

Validation Criteria:

log.rotateSize = 100000

log.keepOld = 10

tools.setinfo.sizeLimit = 1048576

VM04

VM03

SRV01

FW01

log.rotateSize = "100000"

log.keepOld = "3"

1.1.8.

Test Description: Verify Guest Non-Persistent Disks

Test Command: cat guestvmxfile.vmx | grep scsi | grep .mode

Verify values:

Default Value (blank) indicates independant-nonpersistent mode

Blank or "independant-nonpersistent" indicates compliance.

VM04

Rob VandenBrink. rvandenbrink@metafore.ca

COMPLIANT

VM03

COMPLIANT

SRV01

COMPLIANT

FW01

COMPLIANT

1.1.9.

Test Description: Guest Unauthorized device connection

Test Detail: Ensure that unauthorized devices are not currently connected to guest VMs

Test Command: `cat guestvmxfile.vmx | grep .present`

Verify values: Varies by organization

VM04

`floppy0.present = "false"`

`ide0:0.present = "true"`

`ethernet0.present = "true"`

VM03

`floppy0.present = "false"`

`ide0:0.present = "true"`

`ethernet0.present = "true"`

SRV01

`floppy0.present = "true"`

`ethernet0.present = "true"`

FW01

`floppy0.present = "true"`

`ethernet0.present = "true"`

`ethernet1.present = "true"`

`ethernet2.present = "true"`

1.1.10.

Test Description: Guest Prevent Unauthorized Removal or Connection of Devices

Test Command: For each VM: `cat | grep isolation.tools.disable`

Verify Values: should be set to "true". false or blank indicates non-compliance.

VM04

NONCOMPLIANT

Noncompliant settings (blank line indicates noncompliance):

VM03

NONCOMPLIANT

Noncompliant settings (blank line indicates noncompliance):

Rob VandenBrink. rvandenbrink@metafore.ca

SRV01

NONCOMPLIANT

Noncompliant settings (blank line indicates noncompliance):

FW01

NONCOMPLIANT

Noncompliant settings (blank line indicates noncompliance):

1.1.11.

Test Description: Guest Denied Disk Operations D.O.S.

Test Command: `cat /vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9/FW01/FW01.vmx | grep isolation\.tools\.disk`

Verify Values:: should be set to "true". "false" or blank indicates non-compliance.

VM04

VM03

SRV01

FW01

1.1.12.

Test Description: Verify that Guest OS Matches ESX Configuration

Test Command: `cat .vmx | grep guestOS | sort -r`

Verify Values:: should be the same value as the actual OS of the Guest VM.

VM04

guestOS = "linux"

guestOSAltName = "Other Linux (32-bit)"

VM03

guestOS = "linux"

guestOSAltName = "Other Linux (32-bit)"

SRV01

guestOS = "linux"

guestOSAltName = "Other Linux (32-bit)"

FW01

guestOS = "linux"

guestOSAltName = "Other Linux (32-bit)"

1.1.13.

Test Description: Guest File Permissions

Rob VandenBrink. rvandenbrink@metafore.ca

Test Command: ls -lh .vmx

Verify Values:: user and group should be root and root
permissions should be -rwxr-xr-x (755)

```

VM04
-rwxr-xr-x  1 root    root          2.1K Mar  2  2009 /vmfs/volumes/4907c9e2-
e2e30320-806d-000c29dba8b9/VM04/VM04.vmx
VM03
-rwxr-xr-x  1 root    root          2.0K Mar  2  2009 /vmfs/volumes/4907c9e2-
e2e30320-806d-000c29dba8b9/VM03/VM03.vmx
SRV01
-rwxr-xr-x  1 root    root          1.8K Mar  2  2009 /vmfs/volumes/4907c9e2-
e2e30320-806d-000c29dba8b9/SRV01/SRV01.vmx
FW01
-rwxr-xr-x  1 root    root          2.2K Mar  2  2009 /vmfs/volumes/4907c9e2-
e2e30320-806d-000c29dba8b9/FW01/FW01.vmx
list VMDK file permissions
Verify Values:: user and group should be root and root
permissions should be -rw-----

```

```

VM04
VM03
SRV01
FW01

```

1.2. SERVICE CONSOLE checks

1.2.1. _____

Test Description: Firewall Checks

Test Command: esxcfg-firewall -q

Verify Values: Firewall settings should meet corporate standards

```

Chain INPUT (policy DROP 1033 packets, 99877 bytes)
  pkts bytes target     prot opt in     out     source           destination
 316K 218M ACCEPT     all  --  lo     *        0.0.0.0/0        0.0.0.0/0
66486 5989K valid-tcp-flags  tcp  --  *      *        0.0.0.0/0
0.0.0.0/0
66486 5989K valid-source-address !udp --  *      *        0.0.0.0/0
0.0.0.0/0
1033 99877 valid-source-address-udp  udp  --  *      *        0.0.0.0/0
0.0.0.0/0
  12   576 valid-source-address  tcp  --  *      *        0.0.0.0/0
0.0.0.0/0          tcp flags:0x16/0x02
  0     0 icmp-in    icmp --  *      *        0.0.0.0/0        0.0.0.0/0
66474 5988K ACCEPT     all  --  *      *        0.0.0.0/0        0.0.0.0/0
state RELATED,ESTABLISHED

```

Rob VandenBrink. rvandenbrink@metafore.ca

```

    0      0 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:902 state NEW
    0      0 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:80 state NEW
    0      0 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:443 state NEW
    0      0 ACCEPT      udp  --  *    *    0.0.0.0/0      0.0.0.0/0
udp spts:67:68 dpts:67:68
    0      0 ACCEPT      udp  --  *    *    0.0.0.0/0      0.0.0.0/0
udp dpt:427
    0      0 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:427 state NEW
    0      0 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:5989 state NEW
    12    576 ACCEPT      tcp  --  *    *    0.0.0.0/0      0.0.0.0/0
tcp dpt:22 state NEW

```

Chain FORWARD (policy DROP 0 packets, 0 bytes)

pkts	bytes	target	prot	opt	in	out	source	destination
------	-------	--------	------	-----	----	-----	--------	-------------

Chain OUTPUT (policy DROP 0 packets, 0 bytes)

pkts	bytes	target	prot	opt	in	out	source	destination
316K	218M	ACCEPT	all	--	*	lo	0.0.0.0/0	0.0.0.0/0
69644	8959K	valid-tcp-flags	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	icmp-out	icmp	--	*	*	0.0.0.0/0	0.0.0.0/0
112	6408	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
69644	8959K	ACCEPT	all	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
59	4484	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
5982	562K	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
0	0	ACCEPT	tcp	--	*	*	0.0.0.0/0	0.0.0.0/0
189	14426	ACCEPT	udp	--	*	*	0.0.0.0/0	0.0.0.0/0

```

    0      0 ACCEPT      tcp  --  *      *      0.0.0.0/0      0.0.0.0/0
tcp dpt:12345
    0      0 REJECT      all  --  *      *      0.0.0.0/0      0.0.0.0/0
reject-with icmp-port-unreachable

Chain icmp-in (1 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 ACCEPT      icmp -- *      *      0.0.0.0/0      0.0.0.0/0
icmp type 0
    0      0 ACCEPT      icmp -- *      *      0.0.0.0/0      0.0.0.0/0
icmp type 8
    0      0 ACCEPT      icmp -- *      *      0.0.0.0/0      0.0.0.0/0
icmp type 3 code 4
    0      0 DROP       all  -- *      *      0.0.0.0/0      0.0.0.0/0

Chain icmp-out (1 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 ACCEPT      icmp -- *      *      0.0.0.0/0      0.0.0.0/0
icmp type 8
    0      0 ACCEPT      icmp -- *      *      0.0.0.0/0      0.0.0.0/0
icmp type 0
    0      0 DROP       all  -- *      *      0.0.0.0/0      0.0.0.0/0

Chain log-and-drop (7 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 LOG        all  -- *      *      0.0.0.0/0      0.0.0.0/0
LOG flags 6 level 7
    0      0 DROP       all  -- *      *      0.0.0.0/0      0.0.0.0/0

Chain valid-source-address (2 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 DROP       all  -- *      *      127.0.0.1      0.0.0.0/0
    0      0 DROP       all  -- *      *      0.0.0.0/8      0.0.0.0/0
    0      0 DROP       all  -- *      *      0.0.0.0/0
255.255.255.255

Chain valid-source-address-udp (1 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 DROP       all  -- *      *      127.0.0.1      0.0.0.0/0
    0      0 DROP       all  -- *      *      0.0.0.0/8      0.0.0.0/0

Chain valid-tcp-flags (2 references)
  pkts bytes target      prot opt in      out      source      destination
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x3F/0x00
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x11/0x01
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x18/0x08
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x30/0x20
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x03/0x03
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x06/0x06
    0      0 log-and-drop tcp -- *      *      0.0.0.0/0      0.0.0.0/0
tcp flags:0x05/0x05

```

Rob VandenBrink. rvandenbrink@metafore.ca

Incoming and outgoing ports blocked by default.

Enabled services: CIMSLP ntpClient VCB CIMHttpsServer vpxHeartbeats
LicenseClient sshServer

Opened ports:

```
syslog                : port 514 tcp.out udp.out
SEC557Backdoor       : port 12345 tcp.out
```

1.2.2.

Test Description: Running Service Checks

Test Command: chkconfig --list

Verify Values:: Running Services should meet corporate standards

rdisc	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
gpm	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
vmware-webAccess	0:off	1:off	2:off	3:on	4:off	5:off	6:off	6:off
ntpd	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
kudzu	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
syslog	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
microcode_ctl	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
netfs	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
network	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
random	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
rawdevices	0:off	1:off	2:off	3:on	4:on	5:on	6:off	
saslauthd	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
irqbalance	0:off	1:off	2:off	3:on	4:on	5:on	6:off	
iptables	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
smartd	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
snmptrapd	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
sshd	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
portmap	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
nfs	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
nfslock	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
winbind	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
crond	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
xinetd	0:off	1:off	2:off	3:on	4:on	5:on	6:off	
snmpd	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
firewall	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
yum	0:off	1:off	2:off	3:off	4:off	5:off	6:off	
ipmi	0:off	1:off	2:on	3:on	4:on	5:on	6:off	
vmware	0:off	1:off	2:off	3:on	4:off	5:off	6:off	
vmware-vmkauthd	0:off	1:off	2:off	3:on	4:off	5:off	6:off	
vmware-late	0:off	1:off	2:off	3:on	4:off	5:off	6:off	
mptctlnode	0:off	1:off	2:off	3:on	4:off	5:on	6:off	
megaraid_sas_ioctl	0:off	1:off	2:off	3:on	4:off	5:on	6:off	6:off
pegasus	0:off	1:off	2:off	3:on	4:off	5:on	6:off	
wsman	0:off	1:off	2:off	3:on	4:off	5:on	6:off	
mgmt-vmware	0:off	1:off	2:off	3:on	4:off	5:off	6:off	

Rob VandenBrink. rvandenbrink@metafore.ca

```

vmware-autostart    0:off  1:off  2:off  3:on   4:off  5:off  6:off
vmware-vpxa        0:off  1:off  2:off  3:on   4:on   5:on   6:off
xinetd based services:
  chargen-udp:    off
  chargen:        off
  daytime-udp:   off
  daytime:        off
  echo-udp:       off
  echo:           off
  vmware-authd:  on
  services:      off
  time:           off
  time-udp:      off

```

1.2.3.

Test Description: List Running Processes

Test Command: ps -efw --forest

Verify Values:: Check process list for unexpected processes

```

UID          PID    PPID  C  STIME TTY          TIME CMD
root          1      0  0  Sep08 ?           00:00:04 init
root          2      1  0  Sep08 ?           00:00:00 [keventd]
root          3      1  0  Sep08 ?           00:00:00 [ksoftirqd/0]
root          6      1  0  Sep08 ?           00:00:00 [bdf flush]
root          4      1  0  Sep08 ?           00:00:00 [kswapd]
root          5      1  0  Sep08 ?           00:00:03 [kscand]
root          7      1  0  Sep08 ?           00:00:00 [kupdated]
root         19      1  0  Sep08 ?           00:00:00 [vmkmsgd]
root         20      1  0  Sep08 ?           00:00:00 [vmnixhbd]
root         24      1  0  Sep08 ?           00:00:00 [vmkdevd]
root         25      1  0  Sep08 ?           00:00:00 [scsi_ah_0]
root         558     1  0  Sep08 ?           00:00:00 [scsi_ah_1]
root         614     1  0  Sep08 ?           00:00:20 [kjournald]
root         701     1  0  Sep08 ?           00:00:00 [kjournald]
root        1213     1  0  Sep08 ?           00:00:03 syslogd -m 0
root        1217     1  0  Sep08 ?           00:00:00 klogd -x
root        1282     1  0  Sep08 ?           00:00:00 /usr/sbin/sshd
root        3327    1282  0  Sep09 ?           00:00:15  \_ sshd: root@pts/0
root        3329    3327  0  Sep09 pts/0    00:00:09  | \_ -bash
root        27202   3329  0  02:11 pts/0    00:00:00  | \_ /bin/bash ./vm9e.sh
root        27568  27202  0  02:11 pts/0    00:00:00  | \_ ps -efw --forest
root        3598    1282  0  Sep09 ?           00:00:05  \_ sshd: root@pts/1
root        3600    3598  0  Sep09 pts/1    00:00:05  \_ -bash
root         1337     1  0  Sep08 ?           00:00:00 /usr/sbin/vmklogger
root         1381     1  0  Sep08 ?           00:00:03 xinetd -stayalive -pidfile
/var/run/xinetd.pid
ntp          1396     1  0  Sep08 ?           00:00:02 ntpd -U ntp -p /var/run/ntpd.pid
-g
root         1405     1  0  Sep08 ?           00:00:00 gpm -t imps2 -m /dev/mouse
root         1426     1  0  Sep08 ?           00:00:00 /bin/sh /usr/bin/vmware-watchdog
-s webAccess -u 30 -q 5 /usr/lib/vmware/webAccess/j

```

Rob VandenBrink. rvandenbrink@metafore.ca

```

root      1433  1426  0 Sep08 ?           00:00:38  \_
/usr/lib/vmware/webAccess/java/jre1.5.0_15/bin/webAccess -server -Xincgc -Djava.
root      1453      1  0 Sep08 ?           00:00:00  crond
root      1464      1  0 Sep08 ?           00:00:01  /usr/lib/vmware/bin/vmkload_app
--setuid --sched.group=host/vim/vmkauthd --sched.mem
root      1485      1  0 Sep08 ?           00:00:00  /bin/sh /usr/bin/vmware-watchdog
-s hostd -u 60 -q 5 -c /usr/sbin/vmware-hostd-suppo
root      1504  1485  0 Sep08 ?           00:14:34  \_
/usr/lib/vmware/hostd/vmware-hostd /etc/vmware/hostd/config.xml -u
root      1486      1  0 Sep08 ?           00:00:00  logger -t VMware[init] -p
daemon.err
root      1522      1  0 Sep08 ?           00:00:00  /bin/sh /usr/bin/vmware-watchdog
-s cimserver -u 60 -q 5 /var/pegasus/bin/cimserver
root      1529  1522  0 Sep08 ?           00:06:42  \_ /var/pegasus/bin/cimserver
daemon=false
root      1656  1529  0 Sep08 ?           00:00:04  \_
/var/pegasus/bin/cimprovagt 14 17 SLPPProviderModule
root      1728  1529  0 Sep08 ?           00:00:00  \_
/var/pegasus/bin/cimprovagt 18 23 VICimProvider
root      1744  1529  0 Sep08 ?           00:00:00  \_
/var/pegasus/bin/cimprovagt 17 42 VMwareIpmiOemExtenservProvider
root      1928  1529  0 Sep08 ?           00:00:56  \_
/var/pegasus/bin/cimprovagt 10 13 SensorProvider
root      1933  1529  0 Sep08 ?           00:00:42  \_
/var/pegasus/bin/cimprovagt 72 75 RawIpmiProvider
root      1949  1529  0 Sep08 ?           00:00:19  \_
/var/pegasus/bin/cimprovagt 75 78 VMware_LatchedHealthState
root      1955  1529  0 Sep08 ?           00:00:56  \_
/var/pegasus/bin/cimprovagt 79 82 SMBIOSProvider
root      1967  1529  0 Sep08 ?           00:00:18  \_
/var/pegasus/bin/cimprovagt 83 86 LogicalIpmiProvider
root      1971  1529  0 Sep08 ?           00:00:06  \_
/var/pegasus/bin/cimprovagt 87 90 VMware_PortController
root      1978  1529  0 Sep08 ?           00:00:12  \_
/var/pegasus/bin/cimprovagt 90 93 VMware_Controller
root      1980  1529  0 Sep08 ?           00:00:14  \_
/var/pegasus/bin/cimprovagt 93 96 VMware_Battery
root      1982  1529  0 Sep08 ?           00:00:15  \_
/var/pegasus/bin/cimprovagt 96 99 VMware_StorageVolume
root      1985  1529  0 Sep08 ?           00:00:15  \_
/var/pegasus/bin/cimprovagt 99 102 VMware_StorageExtent
root      1988  1529  0 Sep08 ?           00:00:07  \_
/var/pegasus/bin/cimprovagt 102 105 VMware_DiskDrive
root      1992  1529  0 Sep08 ?           00:00:14  \_
/var/pegasus/bin/cimprovagt 105 108 VMware_SASSATAPort
root      1994  1529  0 Sep08 ?           00:00:05  \_
/var/pegasus/bin/cimprovagt 108 111 VMware_StoragePool
root      1996  1529  0 Sep08 ?           00:00:00  \_
/var/pegasus/bin/cimprovagt 112 115 VMware_StoreLibSoftwareIdentity
root      1581      1  0 Sep08 ?           00:00:00  /bin/sh
/opt/vmware/vpxa/bin/vmware-watchdog -s vpxa -u 30 -q 5 /opt/vmware/vpxa/sbi
root      1601  1581  0 Sep08 ?           00:00:07  \_ /opt/vmware/vpxa/vpx/vpxa
root      1632      1  0 Sep08 ?           00:00:00  /bin/sh /usr/bin/vmware-watchdog
-s openwsmand -u 60 -q 5 /sbin/openwsmand -d
root      1647  1632  0 Sep08 ?           00:00:02  \_ /sbin/openwsmand -d
root      1637      1  0 Sep08 tty1         00:00:00  /sbin/mingetty tty1
root      1638      1  0 Sep08 ?           00:00:00  login -- root

```

Rob VandenBrink. rvandenbrink@metafore.ca


```

root      6735  1638  0 Sep09 tty2      00:00:00  \_ -bash
root      1639    1  0 Sep08 tty3      00:00:00  /sbin/mingetty tty3
root      1640    1  0 Sep08 tty4      00:00:00  /sbin/mingetty tty4
root      1641    1  0 Sep08 tty5      00:00:00  /sbin/mingetty tty5
root      1642    1  0 Sep08 tty6      00:00:00  /sbin/mingetty -f
/etc/issue.emergency -1 /bin/login.emergency tty6

```

1.2.4.

Test Description: Control of SSH Root Privileges

Test Command: `Cat /etc/ssh/sshd_config | grep Root`
Verify Values:: PermitRootLogin should be set to no

PermitRootLogin yes

1.2.5.

Test Description: Verify that ftp is disabled by port

Test Command: `netstat -na | grep LIST | grep \:21`
Verify Values:: output should be blank

1.2.6.

Test Description

: Verify that ftp is disabled by configuration
Test Command: `ls -l /etc/xinetd.d | grep ftp`
Verify Values:: output should be blank

1.2.7.

Test Description

: verify that root cannot login directly to the console
Test Command: `cat /etc/securetty`
Verify Values:: this listing should be empty

Test Results:
COMPLIANT

Rob VandenBrink. rvandenbrink@metafore.ca

Raw Data:

```

console
vc/1
vc/2
vc/3
vc/4
vc/5
vc/6
vc/7
vc/8
vc/9
vc/10
vc/11
tty1
tty2
tty3
tty4
tty5
tty6
tty7
tty8
tty9
tty10
tty11

```

1.2.8.

Test Description

: verify su permissions are restricted
 enumerate members of the wheel group

Test Command: cat /etc/group | grep wheel

Verify Value: root only, or authorized users only

```
wheel:x:10:root
```

1.2.9.

Test Description: Verify that only the wheel group members have access to su

Test Command: cat /etc/pam.d/su | grep auth | grep pam_wheel.so | grep -v \#

Test Results:

COMPLIANT

Raw Data:

1.2.10.**Test Description**

: verify that directory service is used for sudo authentication

Test Command: cat /etc/pam.d/sudo

Verify Value: should read "service=system-auth"

COMPLIANT

Raw Data:

```
auth      required    pam_stack.so service=system-auth
account   required    pam_stack.so service=system-auth
password  required    pam_stack.so service=system-auth
```

1.2.11.**Test Description: verify that password aging is enabled**

Test Command: cat /etc/passwd | cust -d : -f 1 (get userids)
chage -l (for each userid, check age)

```
=====
Userid = root
Minimum:      0
Maximum:     -1
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never
=====
Userid = bin
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never
=====
Userid = daemon
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never
=====
```

Rob VandenBrink. rvandenbrink@metafore.ca

```

Userid = adm
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
=====

```

```

Userid = lp
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
=====

```

```

Userid = sync
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
=====

```

```

Userid = shutdown
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
=====

```

```

Userid = halt
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
=====

```

```

Userid = mail
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires: Never

```

```

Password Inactive:      Never
Account Expires:       Never
=====
Userid = news
Minimum:                0
Maximum:                99999
Warning:                7
Inactive:               -1
Last Change:           Oct 10, 2008
Password Expires:      Never
Password Inactive:     Never
Account Expires:       Never
=====
Userid = uucp
Minimum:                0
Maximum:                99999
Warning:                7
Inactive:               -1
Last Change:           Oct 10, 2008
Password Expires:      Never
Password Inactive:     Never
Account Expires:       Never
=====
Userid = operator
Minimum:                0
Maximum:                99999
Warning:                7
Inactive:               -1
Last Change:           Oct 10, 2008
Password Expires:      Never
Password Inactive:     Never
Account Expires:       Never
=====
Userid = gopher
Minimum:                0
Maximum:                99999
Warning:                7
Inactive:               -1
Last Change:           Oct 10, 2008
Password Expires:      Never
Password Inactive:     Never
Account Expires:       Never
=====
Userid = ftp
Minimum:                0
Maximum:                99999
Warning:                7
Inactive:               -1
Last Change:           Oct 10, 2008
Password Expires:      Never
Password Inactive:     Never
Account Expires:       Never
=====
Userid = nobody
Minimum:                0
Maximum:                99999
Warning:                7

```

```

Inactive:      -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```

Userid = nscd
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:     -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```

Userid = vcsa
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:     -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```

Userid = ntp
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:     -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```

Userid = sshd
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:     -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```

Userid = rpc
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:     -1
Last Change:   Oct 10, 2008
Password Expires:  Never
Password Inactive:  Never
Account Expires:  Never

```

```
=====
```

```
Userid = rpcuser
```

```

Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never

```

```

=====
Userid = nfsnobody
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never

```

```

=====
Userid = pcap
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never

```

```

=====
Userid = rpm
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never

```

```

=====
Userid = vimuser
Minimum:      0
Maximum:     99999
Warning:      7
Inactive:    -1
Last Change:      Oct 10, 2008
Password Expires:  Never
Password Inactive: Never
Account Expires:  Never

```

```

=====
Userid = vpxuser
Minimum:      0
Maximum:     -1
Warning:      7
Inactive:    -1
Last Change:      May 28, 2009
Password Expires:  Never
Password Inactive: Never

```

```
Account Expires:      Never
=====
Userid = esx01admin
Minimum:             0
Maximum:             60
Warning:             10
Inactive:            -1
Last Change:         Jan 02, 2009
Password Expires:    Mar 03, 2009
Password Inactive:   Never
Account Expires:     Never
```

1.2.12.

Test Description: Verify Account Lockout

Test Command: `cat /etc/pam.d/system-auth | grep pam_tally`
Verify Values::
 echo for auth required: /lib/security/pam_tally.so no_magic_root
 echo for account required: /lib/security/pam_tally.so deny=3 no_magic_root
 No returned value indicates non-compliance

NONCOMPLIANT

Raw Data:

1.2.13.

Test Description: Verify logging of failed logins

Test Command: `ls /var/log/faillog | wc -l`
Verify Values:: 1 file exists, compliant
 0 file does not exist, noncompliant

COMPLIANT

1.3. MAINTAINING PROPER LOGGING

1.3.1.

Test Description: Verify that NTP is running?

Test Command: `ps -efw | grep ntpd | wc -l`
Verify Values:: 1 = ntpd is running, compliant
 0 = ntpd is not running, noncompliant

COMPLIANT

Rob VandenBrink. rvandenbrink@metafore.ca

1.3.2.

Test Description: Verify that NTP is associated with a trusted timesource?**Test Command:** ntpq associations**Verify Value:** trusted timesource(s) configured
active ntp associations
firewall permissions show ntpClientntp.conf server settings:
server 192.168.206.204
ntp/step-tickers server settings:
server 192.168.206.204

Service Startup Settings

ntpd 0:off 1:off 2:on 3:on 4:on 5:on 6:off

Firewall Settings

NTP Associations:

remote	refid	st	t	when	poll	reach	delay	offset	jitter
ntp01	0.0.0.0	16	u	-	1024	0	0.000	0.000	4000.00

1.3.3.

Test Description: are local logs being written?**Test Command:** date ; echo ; ls -lhst /var/log | head -25**Verify Values::** Compare file date against current date
compare file dates against current date

Current Date is: Thu Sep 10 02:11:48 EDT 2009

```
total 2.8M
248K -rw----- 1 root root 240K Sep 10 02:11 secure
 60K -rw----- 1 root root 53K Sep 10 02:11 messages
 32K -rw----- 1 root root 31K Sep 10 02:04 vmkernel
 76K -rw----- 1 root root 69K Sep 10 02:01 vmksummary
4.0K -rw----- 1 root root 2.6K Sep 10 02:01 cron
4.0K drwxr-xr-x 5 root root 4.0K Sep 9 23:36 vmware
48K -rw-rw-r-- 1 root utmp 44K Sep 9 21:11 wtmp
28K -r----- 1 root root 143K Sep 9 13:29 lastlog
 0 -rw-r--r-- 1 root root 0 Sep 9 05:01 rpmpkgs
4.0K -rw-r--r-- 1 root root 1.3K Sep 9 04:02 vmksummary.txt
8.0K -rw-r--r-- 1 root root 7.9K Sep 9 04:02 rpmpkgs.1
4.0K -rw-r--r-- 1 root root 1.7K Sep 9 04:02 vmksummary.html
8.0K -rw----- 1 root root 7.5K Sep 8 19:59 boot.log
4.0K drwxr-xr-x 2 root root 4.0K Sep 8 19:58 oldconf
 76K -rw----- 1 root root 68K Sep 8 19:58 ksyms.0
 20K -rw-r--r-- 1 root root 17K Sep 8 19:58 dmesg
```

Rob VandenBrink. rvandenbrink@metafore.ca

```

4.0K -rw-----  1 root    root      597 Sep  8 17:35 vmkwarning
   0 -rw-----  1 root    root         0 Sep  8 16:01 maillog
   0 -rw-----  1 root    root         0 Sep  8 16:01 spooler
4.0K -rw-----  1 root    root      846 Sep  8 16:01 cron.1
132K -rw-----  1 root    root     127K Sep  8 15:58 messages.1
 20K -rw-----  1 root    root     18K Sep  8 15:49 secure.1
 28K -rw-----  1 root    root     24K Sep  8 15:47 boot.log.1
 76K -rw-----  1 root    root     68K Sep  8 15:46 ksyms.1

```

1.3.4.

Test Description:

Verify that log file growth is controlled

Test Command: cat /etc/logrotate.conf

Test Results:

```

Size limit:      size 200k
Filename vmkernel Compression Check: /var/log/vmkernel{
    create 0600 root root
    missingok
    nocompress
    # keep a history over 3 years.
    monthly
    rotate 36
    # max log size of 200k (thus limiting total disk usage to under 8megs)
    size 200k
    sharedscripts
    postrotate
        /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null ||
true
    endscrip
}
Noncompliant
Size limit:      size 200k
Filename vmksummary Compression Check: /var/log/vmksummary{
    create 0600 root root
    missingok
    nocompress
    # keep a history over 3 years.
    monthly
    rotate 36
    # max log size of 200k (thus limiting total disk usage to under 8megs)
    size 200k
    olddir /var/log/vmksummary.d
    sharedscripts
    postrotate
        /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null ||
true
    endscrip
}
Noncompliant
Size limit:      size 200k
Filename vmkwarning Compression Check: /var/log/vmkwarning{
    create 0600 root root

```

Rob VandenBrink. rvandenbrink@metafore.ca

```

missingok
sharedscripts
postrotate
    /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null ||
true
    endscript
}
Noncompliant

```

1.3.5.

Test Description: is remote syslog running?

Test Command: cat/etc/syslog.conf | grep @
Test Verify: view syslog.conf, verify remote logging \(@ symbol\
Test Results:

COMPLIANT

Test Raw Output:
*.info;mail.none;authpriv.none;cron.none;local6.none;local5.none
@192.168.206.204

1.3.6.

Test Description: Verify correct permissions on VMware utility file permissions

Test Command: ls -lh /usr/sbin/esxcfg-*
Verify Value: all /usr/sbin/esxcfg-* commands are set to -r-x----- (500)
except esxcfg-auth which should be -r-xr--r-- (544)
Test Results:

esxcfg* permissions - COMPLIANT

Test Results: - n esxcfg-auth permissions -
COMPLIANT

Test Raw Data:

```

-r-x----- 1 root    root      48K Aug 12  2008 /usr/sbin/esxcfg-advcfg
-r-xr--r-- 1 root    root      26K Aug 12  2008 /usr/sbin/esxcfg-auth
-r-x----- 1 root    root      19K Aug 12  2008 /usr/sbin/esxcfg-boot
-r-x----- 1 root    root     7.6K Aug 12  2008 /usr/sbin/esxcfg-
configcheck
-r-x----- 1 root    root     51K Aug 12  2008 /usr/sbin/esxcfg-
dumppart
-r-x----- 1 root    root     29K Aug 12  2008 /usr/sbin/esxcfg-
firewall
-r-x----- 1 root    root     2.9K Aug 12  2008 /usr/sbin/esxcfg-hwiscsi
-r-x----- 1 root    root     42K Aug 12  2008 /usr/sbin/esxcfg-info
-r-x----- 1 root    root     19K Aug 12  2008 /usr/sbin/esxcfg-init

```

Rob VandenBrink. rvandenbrink@metafore.ca

```

-r-x----- 1 root    root      11K Aug 12 2008 /usr/sbin/esxcfg-
linuxnet
-r-x----- 1 root    root      42K Aug 12 2008 /usr/sbin/esxcfg-module
-r-x----- 1 root    root      89K Aug 12 2008 /usr/sbin/esxcfg-mpath
-r-x----- 1 root    root      53K Aug 12 2008 /usr/sbin/esxcfg-nas
-r-x----- 1 root    root      48K Aug 12 2008 /usr/sbin/esxcfg-nics
-r-x----- 1 root    root      24K May  5 2008 /usr/sbin/esxcfg-pciid
-r-x----- 1 root    root      5.6K Aug 12 2008 /usr/sbin/esxcfg-rescan
-r-x----- 1 root    root      40K Aug 12 2008 /usr/sbin/esxcfg-resgrp
-r-x----- 1 root    root      49K Aug 12 2008 /usr/sbin/esxcfg-route
-r-x----- 1 root    root      59K Aug 12 2008 /usr/sbin/esxcfg-swiscsi
-r-x----- 1 root    root      9.2K Aug 12 2008 /usr/sbin/esxcfg-upgrade
-r-x----- 1 root    root      53K Aug 12 2008 /usr/sbin/esxcfg-
vmhbadevs
-r-x----- 1 root    root      57K Aug 12 2008 /usr/sbin/esxcfg-vmknic
-r-x----- 1 root    root      66K Aug 12 2008 /usr/sbin/esxcfg-vswif
-r-x----- 1 root    root      84K Aug 12 2008 /usr/sbin/esxcfg-vswitch

```

1.3.7.

Test Description: Verify correct file permissions on logs

Test Commands

```

ls -lh /var/log
ls -lh /var/log/vmware
ls -lh /var/log
ls -Rlh /vmfs/volumes/

```

Verify Values:: all log files have permissions of -rw----- (600)
 except: /var/log/vmware/webAccess/* -rw-r--r-- (755)
 and: Individual Virtual Machine log files -rw-r--r-- (644)

Test Results:

in /var/log:

Test Raw Data:

```

-rw----- 1 root    root      7.5K Sep  8 19:59 boot.log
-rw----- 1 root    root      24K Sep  8 15:47 boot.log.1
-rw----- 1 root    root      8.2K Sep  4 14:00 boot.log.2
-rw----- 1 root    root      34K May 27 20:10 boot.log.3
-rw----- 1 root    root      16K Apr  5 00:58 boot.log.4
-rw----- 1 root    root      2.6K Sep 10 02:01 cron
-rw----- 1 root    root      846 Sep  8 16:01 cron.1
-rw----- 1 root    root      2.0K Sep  4 14:01 cron.2
-rw----- 1 root    root      1.3K May 27 21:01 cron.3
-rw----- 1 root    root      414 Apr  5 01:01 cron.4
-rw----- 1 root    root      68K Sep  8 19:58 ksyms.0
-rw----- 1 root    root      68K Sep  8 15:46 ksyms.1
-rw----- 1 root    root      68K Sep  5 19:37 ksyms.2
-rw----- 1 root    root      68K Sep  4 17:56 ksyms.3
-rw----- 1 root    root      68K Sep  4 13:59 ksyms.4
-rw----- 1 root    root      68K May 27 20:08 ksyms.5

```

Rob VandenBrink. rvandenbrink@metafore.ca

```

-rw----- 1 root root 68K May 17 07:04 ksyms.6
-rw----- 1 root root 0 Sep 8 16:01 maillog
-rw----- 1 root root 0 Sep 4 14:01 maillog.1
-rw----- 1 root root 0 May 27 21:01 maillog.2
-rw----- 1 root root 0 Apr 5 01:01 maillog.3
-rw----- 1 root root 0 Mar 29 00:01 maillog.4
-rw----- 1 root root 53K Sep 10 02:11 messages
-rw----- 1 root root 127K Sep 8 15:58 messages.1
-rw----- 1 root root 820 Dec 31 2008 messages
@192.168.206.136
-rw----- 1 root root 4.9K Dec 31 2008 messages
@192.168.206.204
-rw----- 1 root root 40K Sep 4 14:01 messages.2
-rw----- 1 root root 177K May 27 20:52 messages.3
-rw----- 1 root root 83K Apr 5 01:01 messages.4
-rw----- 1 root root 240K Sep 10 02:11 secure
-rw----- 1 root root 18K Sep 8 15:49 secure.1
-rw----- 1 root root 1.7K Sep 4 14:01 secure.2
-rw----- 1 root root 10K May 27 20:51 secure.3
-rw----- 1 root root 4.6K Apr 5 01:00 secure.4
-rw----- 1 root root 0 Sep 8 16:01 spooler
-rw----- 1 root root 0 Sep 4 14:01 spooler.1
-rw----- 1 root root 0 May 27 21:01 spooler.2
-rw----- 1 root root 0 Apr 5 01:01 spooler.3
-rw----- 1 root root 0 Mar 29 00:01 spooler.4
-rw----- 1 root root 0 Oct 28 2008 storageMonitor
-rw----- 1 root root 31K Sep 10 02:04 vmkernel
-rw----- 1 root root 200K Sep 5 19:50 vmkernel.1
-rw----- 1 root root 221K Apr 8 12:55 vmkernel.2
-rw----- 1 root root 200K Jan 23 2009 vmkernel.3
-rw----- 1 root root 205K Dec 16 2008 vmkernel.4
-rw----- 1 root root 221K Nov 7 2008 vmkernel.5
-rw----- 1 root root 0 Oct 28 2008 vmkproxy
-rw----- 1 root root 69K Sep 10 02:01 vmksummary
-rw----- 1 root root 597 Sep 8 17:35 vmkwarning
-rw----- 1 root root 4.5K Sep 4 16:23 vmkwarning.1
-rw----- 1 root root 0 May 27 21:01 vmkwarning.2
-rw----- 1 root root 864 May 27 20:45 vmkwarning.3
-rw----- 1 root root 0 Mar 29 00:01 vmkwarning.4

```

In /var/log/vmware/webAccess:

NONCOMPLIANT

Noncompliant files:

total 12K

```

-rw-r--r-- 1 root root 0 Oct 28 2008 objectMonitor.log
-rw-r--r-- 1 root root 9.7K Oct 28 2008 proxy.log
-rw-r--r-- 1 root root 0 Sep 8 19:59 timer.log
-rw-r--r-- 1 root root 0 Sep 8 19:59 unitTest.log
-rw-r--r-- 1 root root 0 Oct 28 2008 updateThread.log
-rw-r--r-- 1 root root 0 Oct 28 2008 viewhelper.log

```

Test Raw Data:

total 12K

```

-rw-r--r-- 1 root root 0 Oct 28 2008 objectMonitor.log
-rw-r--r-- 1 root root 9.7K Oct 28 2008 proxy.log
-rw-r--r-- 1 root root 0 Sep 8 19:59 timer.log
-rw-r--r-- 1 root root 0 Sep 8 19:59 unitTest.log

```

Rob VandenBrink. rvandenbrink@metafore.ca

```
-rw-r--r-- 1 root root 0 Oct 28 2008 updateThread.log
-rw-r--r-- 1 root root 0 Oct 28 2008 viewhelper.log
```

Individual VM Logs

NONCOMPLIANT

Noncompliant files:

Raw Data:

```
-rw-r--r-- 1 root root 19K Dec 16 2008 vmware-18.log
-rw-r--r-- 1 root root 21K Jan 12 2009 vmware-19.log
-rw-r--r-- 1 root root 9.8K Mar 2 2009 vmware-20.log
-rw-r--r-- 1 root root 9.3K Mar 2 2009 vmware.log
-rw-r--r-- 1 root root 18K Nov 13 2008 vmware-14.log
-rw-r--r-- 1 root root 18K Dec 4 2008 vmware-15.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-16.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-17.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-18.log
-rw-r--r-- 1 root root 18K Jan 23 2009 vmware-19.log
-rw-r--r-- 1 root root 17K Mar 2 2009 vmware.log
-rw-r--r-- 1 root root 18K Nov 13 2008 vmware-10.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-11.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-12.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-13.log
-rw-r--r-- 1 root root 18K Nov 12 2008 vmware-8.log
-rw-r--r-- 1 root root 19K Nov 13 2008 vmware-9.log
-rw-r--r-- 1 root root 18K Mar 2 2009 vmware.log
-rw-r--r-- 1 root root 19K Nov 13 2008 vmware-10.log
-rw-r--r-- 1 root root 18K Dec 16 2008 vmware-11.log
-rw-r--r-- 1 root root 19K Dec 16 2008 vmware-12.log
-rw-r--r-- 1 root root 13K Nov 11 2008 vmware-7.log
-rw-r--r-- 1 root root 13K Nov 12 2008 vmware-8.log
-rw-r--r-- 1 root root 13K Nov 13 2008 vmware-9.log
-rw-r--r-- 1 root root 18K Mar 2 2009 vmware.log
-rw-r--r-- 1 root root 37 May 28 18:08 SRV02-d464cf93.hlog
-rw-r--r-- 1 root root 16K May 27 11:22 vmware-50.log
-rw-r--r-- 1 root root 16K May 27 11:31 vmware-51.log
-rw-r--r-- 1 root root 16K May 27 11:36 vmware-52.log
-rw-r--r-- 1 root root 16K May 27 11:48 vmware-53.log
-rw-r--r-- 1 root root 33K May 27 13:26 vmware-54.log
-rw-r--r-- 1 root root 34K May 28 18:08 vmware-55.log
-rw-r--r-- 1 root root 30K May 28 12:00 vmware.log
```

1.3.8.

Test Description: Is root filesystem protected from filling up?

verify that /home, /tmp and /var/log are in 3 separate, non-root partitions

Test Command: vdf -lh

Verification Value: verify a separate entry for each critical partition:
/var/log, /home, tmp

Test Results

/var/log partition is COMPLIANT
/home partition is NONCOMPLIANT
/tmp partition is NONCOMPLIANT

Test Raw Data:

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	2.2G	1.6G	457M	78%	/
none	124M	0	124M	0%	/dev/shm
/dev/sda2	496M	8.1M	462M	2%	/var/logs
/vmfs/devices	11G	0	11G	0%	/vmfs/devices
/vmfs/volumes/4907c9e2-e2e30320-806d-000c29dba8b9	768M	178M	590M	23%	/vmfs/volumes/LABVMFS
/vmfs/volumes/4912fc0b-4723c25b-f882-000c29dba8b9	768M	659M	109M	85%	/vmfs/volumes/LABVMFS2

2. Appendix - Windows Scripts

2.1. Master Inventory Script (invent-list.cmd)

The master script is simply a series of calls, one per server, to invent-host.cmd. Invent-host.cmd is the master inventory script for Windows servers.

```
call inven-host mail.somecompany.com 10.19.1.4
call inven-host dc01.somecompany.com 10.19.1.5
call inven-host finance2.somecompany.com 10.19.1.6
call inven-host dc02.somecompay.com 10.19.1.7
call inven-host auth1.somecompany.com 10.19.1.8
```

The script calls using both the hostname and ip address. The hostname is used in report titles and the like, the ip address is generally used for host access. This approach is used so that DNS is not required to successfully complete a documentation run.

2.2. Windows Server Detail Inventory Script (invent-host.cmd)

set HOST=%1	Set variables, based on the command line parameters
set IP=%2	
if !_%2==_ set IP=%1	Check to ensure that all parameters were passed. If not, set one to the other.
set DIRNAME=%HOST%	
if NOT %HOST%=%IP% set DIRNAME=%HOST%_%IP%	Set a target directory
set DOM=somecompany.com	Domain and credential info. This will be encrypted and moved out of the script in a future release
set UID=%DOM%\adminaccount	
set PWD=AdminPassword	
ping -w 1000 -n 2 %IP% find "TTL=" goto HOSTDOWN	Is the host up?
net use * /d /y	
net use \\%IP%\c\$ /user:%UID% /password:%PWD% && goto HOSTOK	Are the credentials good?
goto BADCREDS	
.HOSTOK	
md %DIRNAME%	
rem combine some, add systemenclosure stuff	
rem wmic startup list full (malware)	
rem printers	
wmic /output:%DIRNAME%\patches.htm /user:%UID% /password:%PWD% /node:%IP% qfe list brief /format:htable	Get patch list (QFE)
wmic /output:%DIRNAME%\os.htm /user:%UID% /password:%PWD% /node:%IP% os list full /format:hform	OS Information
wmic /output:%DIRNAME%\bios.htm /user:%UID% /password:%PWD% /node:%IP% bios list brief /format:htable	BIOS Information
wmic /output:%DIRNAME%\computersystem.htm /user:%UID% /password:%PWD% /node:%IP%	Computer System info

Rob VandenBrink. rvandenbrink@metafore.ca

computersystem list /format:hform	
wmic /output:%DIRNAME%\partitions.htm /user:%UID% /password:%PWD% /node:%IP% logicaldisk list brief /format:htable	Disk partition info
wmic /output:%DIRNAME%\csproduct.htm /user:%UID% /password:%PWD% /node:%IP% csproduct list brief /format:htable	Computer System Product names
wmic /output:%DIRNAME%\groups.htm /user:%UID% /password:%PWD% /node:%IP% group list /format:htable	Local Groups (Domain groups if target is a DC)
wmic /output:%DIRNAME%\users.htm /user:%UID% /password:%PWD% /node:%IP% useraccount list /format:htable	Local users (Domain users if target is a DC)
wmic /output:%DIRNAME%\sysaccounts.htm /user:%UID% /password:%PWD% /node:%IP% sysaccount list /format:htable	System account list
REM wmic /output:%DIRNAME%\p.htm /user:%UID% /password:%PWD% /node:%IP% netclient list /format:htable	
wmic /output:%DIRNAME%\shares.htm /user:%UID% /password:%PWD% /node:%IP% share list /format:htable	Local shares
wmic /output:%DIRNAME%\NICs.htm /user:%UID% /password:%PWD% /node:%IP% nicconfig list /format:htable	NIC Configurations
wmic /output:%DIRNAME%\ACTIVENICs.htm /user:%UID% /password:%PWD% /node:%IP% nicconfig where IPEnabled='true' /format:htable	NIC Configurations of live NICs only
wmic /output:%DIRNAME%\products.htm /user:%UID% /password:%PWD% /node:%IP% product get vendor_name,version,installdate,packagecache,description,identifyingnumber /format:htable	Installed Program List
wmic /output:%DIRNAME%\services.htm /user:%UID% /password:%PWD% /node:%IP% service list /format:htable	Windows Services
rem list out share permissions	
"c:\program files\systemtools\dumpsec" /computer=\\%1 /rpt=allsharedirs /outfile=d:\support.www\scripts\win\reports\lacls\%1_shares.txt /saveas=fixed	Share permissions (using dumpsec)
rem list out filesystem permissions	
rem "c:\program files\systemtools\dumpsec" /computer=\\%1 /rpt=dir=\\%1c\$ /showaudit /outfile=d:\support.www\scripts\win\reports\lacls\%1_c.txt /saveas=fixed	
rem "c:\program files\systemtools\dumpsec" /computer=\\%1 /rpt=dir=\\%1d\$ /showaudit /outfile=d:\support.www\scripts\win\reports\lacls\%1_d.txt /saveas=fixed	
goto ENDEND	
:HOSTDOWN	
date /t >> hosts.err	Maintain an error file for down hosts
time /t >> hosts.err	
echo %1 - host down (no icmp echo reply) >>hosts.err	
goto ENDEND	
:BADCREDS	
date /t >> hosts.err	Also indicate errors for bad credentials
time /t >> hosts.err	
echo %1 - bad credentials or no SMB services >> hosts.err	
:ENDEND	

3. Appendix - Novell Scripts

The novell script takes a more monolithic approach – data is collected on the NDS directory in this script. The server-specific calls simply collect the config.nlm output from each server

3.1. Novell Detail List

cx /r	Go to the root of NDS Tree
rem list NDS tree	
cx /t /c >reports\ndstreelist.txt	List NDS Tree
rem list objects within NDS tree	
cx /t /a /c >reports\ndsobjlist.txt	List all objects in nds in tree format
rem list all users, with expire on logins and passwords	
nlist user=* /s > reports\userids.txt	List all users
rem list accounts that don't require a password	
NLIST User = * WHERE "Require a Password" NEXISTS /s > reports\userids.nopassword.txt	Users who don't require a password.
rem list accounts that haven't logged in for an extended period of time	
NLIST User = * WHERE "Last Login Time" LT 1/11/2008 /s > reports\userids.inactive.txt	List inactive accounts
rem list disabled accounts	
nlist user=* where "Account Disabled" EQ "Yes" /s > reports\userids.disabled.txt	List disabled accounts
NLIST Organization SHOW "Object Trustees (ACL)" /S /c > reports\trustees.o.txt	Show object trustees
NLIST "Organizational Unit" SHOW "Object Trustees (ACL)" /S /c > reports\trustees.ou.txt	Show trustees of organizational units
Rem collect config.txt reports	
call getconfig fileserver	Collect config.txt from each netware server
call getconfig firewall	
call getconfig webhost	
call getconfig zendeploy	
call getconfig zenmaster	
rem show volumes (used by other scripts)	
rem nlist volume /s /c > volumes.txt	Volume list (disabled, it's in the config.txt output)
rem show groups and group members	
nlist group show MEMBER /c/s > reports\grouplist.txt	Group member list
rem volume reports	
call vol1	

3.2. Getconfig.cmd

This script collects config.txt files from Netware Servers.

net use r: /d	Remove any existing drive map
net use r: \\%1\sys	Map the drive
xcopy r:\system\config.txt reports\%1_config.txt /y	
call txt2html fileserver_volumes "Server FILESERVER - Volume ACL Report"	Change all TXT files to HTML
call txt2html fileserver_config "Server FILESERVER - CONFIG.NLM Report"	This entire section needs a rework, as the resulting HTML
call txt2html firewall_config "Server FIREWALL - CONFIG.NLM Report"	Is very clumsy – more use of <PRE> is in order for instance
call txt2html firewall_volumes "Server FIREWALL - Volume ACL Report"	
call txt2html grouplist "NDS Group Report"	
call txt2html webhost_volumes "Server WEBHOST - Volume ACL Report"	
call txt2html ndsobjlist "NDS Object ACL Report"	
call txt2html webhost_config "Server WEBHOST - CONFIG.NLM ACL Report"	
call txt2html ndstreelist "DNS Tree Listing"	
call txt2html trustees.o "NDS Object Trustee Assignments"	
call txt2html trustees.ou "NDS OU Trustee Assignments"	
call txt2html userids.disabled "NDS - Disabled Accounts Report"	
call txt2html userids.inactive "NDS - Inactive Accounts Report"	
call txt2html userids.nopassword "NDS - Accounts with no Password Report"	
call txt2html userids "NDS - USERID Detail Report"	
call txt2html zendeploy_config "Server ZENDEPLOY - CONFIG.NLM Report"	
call txt2html zendeploy_volumes "Server ZENDEPLOY - Volume ACL Report"	
call txt2html zenmaster_config "Server ZENMASTER - CONFIG.NLM Report"	
call txt2html zenmaster_volumes "Server ZENMASTER - Volume ACL Report"	

3.3. Convert Text Data to HTML

This script relies heavily on sed. First the sed file is created, then it is called:

sed s/"Insert Title Here"%2/g txt2html.sed >work1.sed	Take the sed skeleton file and update the Title text
sed -f work1.sed reports\%1.txt >reports\%1.html	Now run the SED using the input file and resulting SED file

3.4. An example WORK1.SED file

s/^\&g	These three lines are unnecessary and
s/^\</g	mess up the resulting HTML file
s/^\>/g	They'll be replaced with a single <PRE>
1 i \	
<head>\	
<title>\	
Server FILESERVER - CONFIG.NLM Report\	
</title>\	
</head>\	
<body>\	
<p>Server FILESERVER - CONFIG.NLM Report</p>\	
<p> </p>\	
<p> </p>\	
<p>\	
<pre>	
\$ a \	
\	
</pre>\	
</body>\	
</html>	

4. Appendix - Front-end Web Page

As mentioned, all IT-DOCS reports are front-ended with a simple website.

At the client's request, the main page matches the customer rack layout, and server hyperlinks are made off the network and rack diagrams.

UPS PDU documentation is currently manual, it will be automated using data collection via telnet and/or ssh (depending on the PDU) in a future version

















The "Directories" hyperlink has sub-pages for Active Directory and NDS.

The Firewall, Switch Topology and VLAN pages are diagrams with further links.

Network Documentation

- [Servers](#)
- [PDU Diagrams](#)
- [rack2](#)
- [vms](#)
- [Firewall](#)
- [Directories](#)
- [Switch Topology](#)
- [Network VLANs](#)

Some Company – Network Documentation

patch1		10.19.1.52
backups		10.19.1.22
auth1		10.19.1.8
sql02		10.191.7
ESX01		10.19.1.6
sql01		10.19.1.12
tbd		10.19.1.7
backupsrv-notes		10.19.1.14
fileserv		10.19.1.17
ESX02		10.19.1.5
ESX03		10.191.4
zendeploy		10.19.1.16
zenmaster		10.19.1.39
citrix1		10.19.1.9
citrix2		10.19.1.19
Firewall		10.19.1.3

© 2010 SANS Institute, Author retains full rights.

Upcoming SANS Penetration Testing



Click Here to
{Get Registered!}



SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS
Mentor Session - SEC504	Boston, MA	Sep 26, 2017 - Nov 07, 2017	Mentor
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
SANS Oslo Autumn 2017	Oslo, Norway	Oct 02, 2017 - Oct 07, 2017	Live Event
SANS vLive - SEC542: Web App Penetration Testing and Ethical Hacking	SEC542 - 201710,	Oct 03, 2017 - Nov 09, 2017	vLive
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
Community SANS Chicago SEC504*	Chicago, IL	Oct 09, 2017 - Oct 14, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Mentor Session - SEC504	Columbia, SC	Oct 10, 2017 - Nov 21, 2017	Mentor
SANS Tysons Corner Fall 2017	McLean, VA	Oct 14, 2017 - Oct 21, 2017	Live Event
Community SANS New York SEC542*	New York, NY	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Tokyo Autumn 2017	Tokyo, Japan	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS Brussels Autumn 2017	Brussels, Belgium	Oct 16, 2017 - Oct 21, 2017	Live Event
SANS vLive - SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking	SEC660 - 201710,	Oct 17, 2017 - Nov 22, 2017	vLive
Community SANS Columbus SEC504	Columbus, OH	Oct 23, 2017 - Oct 28, 2017	Community SANS
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 2017	Live Event
Mentor Session - SEC504	Dayton, OH	Oct 23, 2017 - Nov 27, 2017	Mentor
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
Community SANS Des Moines SEC504*	Des Moines, IA	Oct 30, 2017 - Nov 04, 2017	Community SANS
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, United Arab Emirates	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Miami 2017	Miami, FL	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Milan November 2017	Milan, Italy	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS New York SEC504*	New York, NY	Nov 06, 2017 - Nov 11, 2017	Community SANS
Mentor Session AW - SEC504	Houston, TX	Nov 06, 2017 - Jan 29, 2018	Mentor
SANS Amsterdam 2017	Amsterdam, Netherlands	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS Raleigh SEC504	Raleigh, NC	Nov 06, 2017 - Nov 11, 2017	Community SANS
Community SANS Columbia SEC504	Columbia, MD	Nov 08, 2017 - Nov 15, 2017	Community SANS