Is It Patched Or Is It Not?

GIAC (GSEC) Gold Certification

Author: Jason Simsay, jason.simsay@gmail.com
Advisor: Adam Kliarsky
Accepted: January 23, 2015

Abstract

Patch management tools may produce conflicting results. A single host may be found to be missing a patch by one tool and not missing that same patch by another. The patch management strategy within our organization calls for using one tool to deploy patches and another for validating that patches have been installed. Discrepancies between the tools are frequent enough that using just one tool is clearly not adequate. Is the patch applied or isn't it? Should it be? How do we know? Software vendors answer these questions in their advisories. Patch management vendors compile metadata that contains the logic to determine whether a patch is applicable to and installed on a host. Their tools analyze files, the registry, and host configuration settings against their metadata. The result is a patch applicability and installation status report. A disagreement amongst tools can leave an organization asking the same questions software vendors have already answered.

1. Introduction

1.1. Real World Situation

Organizational changes bring new perspectives. Our organization has a long standing patch reporting process built around a system that essentially leverages the Microsoft Update service to determine compliance with Microsoft security updates. Symantec has maintained the RMS (Risk Management System) Information Server product, since acquiring it from BindView Development in 2006 (Symantec, 2005), as an agentless data collection platform that could stand alone and/or be integrated with Symantec's Control Compliance Suite (CCS) application. The release of CCS version 11 marks the end of life for the RMS Information Server. The RMS Information Server had two options for Microsoft patch assessment. It could use the Shavlik scan engine and patch metadata or it could leverage the Windows Update component and use either Microsoft Update or an existing Windows Server Update Services (WSUS) infrastructure as the patch authority. CCS offers only the Shavlik option.

Interestingly, our organization had formerly used the Shavlik option with RMS. It was abandoned in favor of the Microsoft Update option due to a high number of false positives. At the time WSUS was used for patch deployment. Today, Lumension is used for deployment. Lumension leverages Microsoft's patch metadata contained in the wsusscn2.cab file. Where Lumension and Microsoft have relative harmony, the Shavlik metadata introduces some apparent conflict. In migrating patch compliance reporting to CCS and Shavlik suddenly there are missing patches. How do we resolve discrepancies in patch compliance reports? Is this discrepancy just a hurdle to overcome or is there something more to it?

1.2. Expectations

This paper will explore the claim that different patch management tools produce different results. This exploration will describe the Microsoft security release process, evaluate the results of a comparative analysis using four patch management tools, and consider strategies for resolution of discrepancies. No attempt will be made to identify which tool is right or wrong, better or worse.

1.3. Microsoft Security Updates

The Second Edition of the Microsoft Security Update Guide (MSUG) is a detailed reference guide for information security professionals. The intent of the document is to help organizations manage the process of deploying Microsoft security updates (Microsoft, p. 1). It is not going to shed light on anything conceptually new but it does provide a concise description of Microsoft's security information delivery processes. An organization familiar with the MSUG has the necessary understanding of what to expect from Microsoft and can tune their patch management program accordingly. Microsoft communicates security release information in three ways: the well-known Patch Tuesday delivery of security updates requiring installation on applicable systems, the awe inspiring out-of-band release of a security update to address vulnerabilities deemed too severe to wait for the next Patch Tuesday, and finally the security advisory (Microsoft, p. 26). Unlike the first two, security advisories do not typically have an associated security bulletin. They may or may not lead to the release of a security update.

Prior to the release of any security update package, an advance notification is issued that describes the "aggregate maximum severity" of the planned release. This allows organizations to allocate resources and prepare accordingly so that a minimum of time elapses between the release of the update package and systems being patched (Microsoft, p. 19). Once released, malicious entities can leverage the binary in the update package to further the development of working exploit code. The advanced notification is replaced by a security bulletin summary that defines the maximum potential severity (Microsoft, p. 36) and vulnerability impact, exploitability index, and identifies the affected software for each security bulletin.

Individual security bulletins contain information to assist information security professionals contemplating the risk of the vulnerability within the context of their organization. The summary and individual bulletins dictate what needs to be patched. The Security Update Deployment section provides the information necessary to determine whether an applicable update was successfully installed. The file version information generally refers back to the relevant knowledge base article which lists the updated files and versions that are included in the security update. Registry key verification may or may not be available to validate installation success.

1.4. Predictions

"Microsoft Security Bulletins are your single, authoritative source for the important information you need about Microsoft Security Updates" (Budd, 2006). In other words, the security bulletin contains all the information that is necessary to perform a vulnerability risk analysis, identify and update applicable systems, and validate that the update has been installed successfully.

Microsoft Security Bulletins identify the affected and non-affected software. However, it is not always that simple. MS12-070 describes a vulnerability affecting multiple versions of Microsoft SQL Server. The bulletin makes it clear that the update is only offered to those servers running SQL Server Reporting Services. Looking closer, we notice that the FAQ section states that only instances of SQL server that are not disabled will be updated. The applicability logic starts with the affected software, and in the case of this bulletin we must consider the installed services and the startup type for each instance. MS14-007 is another example of a bulletin with some added complexity in the applicability logic. Windows 7 and Windows 2008 R2 and later systems are affected. However, only those systems with a specific platform update installed are offered the update.

Patch metadata logic is not trivial. It is a combination of occasionally complex applicability logic, installation verification, with still yet other considerations. Patches may have relationships to other patches. A security update may be released that supersedes a prior update. Complicated logic increases the probability of error and it is apparent that not all logic is created equal. Ron Gula, the Chief Technology Officer for Tenable Network Security, blogs about patch audit tool collisions (Gula, 2009^2) and misleading patch audits (Gula, 2009^1). Shavlik knowledgebase articles attempt to explain why Shavlik patch reports differ from Windows Update (Shavlik, 2013).

2. Analysis

2.1 Tools

Determining whether an update is applicable to a system starts with checking for installed software and then checking for additional signs such as the existence of a file version, a directory, configuration setting, file checksum or size, or other information about installed software packages (Keller, p. 2). This information is used by patch management vendors to create vulnerability metadata. Their scan engine uses the metadata to identify applicable systems. Presumably, patch management tools utilize the same vulnerability data, as that which is available in Microsoft security bulletins, to determine which patches apply and which are installed. This analysis will compare the patch reports produced using four patch management tools.

2.1.1. Lumension Endpoint Management and Security Suite – Patch and Remediation

The Patch and Remediation product is a component of the overall Lumension Endpoint Management and Security Suite. The product includes a server and endpoint services. Endpoint agents are available for Windows, MAC, various flavors of Linux and Solaris, AIX and HP-UX. Lumension packages software from numerous vendors including Adobe, Apple, Citrix, Microsoft, Mozilla, Novell, Oracle, SUN, and VMware among others. All vulnerability and package content is retrieved from the Global Subscription Service (GSS). In addition to patches, Lumension can deploy administrative tasks, configuration actions, and custom packages, and can also execute scripts.

Lumension uses agent based technology. Endpoints can be discovered and agents can be deployed from the server based web administration console. Once installed the agents implement a pull technology and are configured to check in with the server at regular intervals. On check in, the server will deploy any packages or tasks that have reached their scheduled deployment time relative to the endpoint. Mandatory baselines can ensure that all endpoints have critical patches installed. If the GSS has updated vulnerability data since the last check in, the Discover Applicable Updates (DAU) task will be deployed to the endpoint.

The DAU task identifies patches and software that are applicable to the endpoint. The system does not provide any apparent information regarding the evidence for patch compliance. Lumension has a proprietary inventory collection scan but also leverages Microsoft's wsusscn2.cab file to identify applicable patches. Vulnerability metadata is updated within about 24 hours following the Microsoft release. Generally, the patch applicability results reported by Lumension will align with those reported by the Microsoft Baseline Security Analyzer (MBSA). If Microsoft does not offer a patch to a given system via Microsoft Update, Lumension will not either. Our organization uses Lumension for patch deployments. Experience has shown the occasional patch to be marked as superseded in Lumension when MBSA reports it to be missing.

2.1.2. Symantec Control Compliance Suite

The Symantec Control Compliance Suite (CCS) is an extensive security management application that addresses many of an organization's risk and regulatory compliance needs. The Standards module provides data collection to identify assets and retrieve evidence data to measure compliance with industry best practices and internal standards (Manzuik, p. 178). Entitlement data collection, policy lifecycle management, ad hoc endpoint query, and reporting and dashboard capabilities are also present. CCS can analyze patch compliance for Microsoft Windows as well as UNIX based assets. Windows patch assessments leverage the Shavlik vulnerability metadata and scan technology to analyze Microsoft and many third party software patches. Shavlik is the "fastest to market with patch data" (Nicastro, figure 7.1). Symantec requires additional time before the patch assessment content update packages are available to CCS customers, typically the Friday following Microsoft's patch Tuesday.

Agentless patch assessments leverage the Server Message Block (SMB) protocol to analyze the file system and registry of Windows hosts. CCS Managers may be distributed within your environment to get them nearer the endpoints but may require firewall allowances. The Shavlik technology reports back the file or registry evidence for missing patches so that administrators can readily understand why a particular patch is reported to be missing. Patch assessment queries are unable to collect data from assets that happen to be offline at the time. An alternative to the query method is a patch assessment standard evaluation. This allows you to leverage a function of the standards data collection module that permits recurring collections to retrieve data from assets missed on previous runs. The standard evaluation also has an integrated exceptions management component allowing you to request and approve patch exceptions by asset.

Our organization uses Control Compliance Suite's repackaged Shavlik technology for patch reporting. Nicastro touts Shavlik as trusted and proven technology (Nicastro, Figure 7.1). File version analysis seems to play heavily in determining patch applicability. At times the finer particulars of system configuration Microsoft takes into consideration when determining applicability, are seemingly discounted by the Shavlik engine. This is interesting given that the predecessor to Shavlik Protect was the "Network Security Hotfix Checker" which was developed by Shavlik Technologies and at one time was distributed by Microsoft (Cole, p. 199)

2.1.3. Microsoft Baseline Security Analyzer

The Microsoft Baseline Security Analyzer (MBSA) is a free tool provided by Microsoft that inspects local or remote system configurations for Windows, IIS, or SQL administrative vulnerabilities, weak passwords, and security updates. It could be described as an agentless technology in that it does not require any additional software other than the Windows Update component that is built into the Windows operating system. Security update analysis can leverage Microsoft Update or a Windows Server Update Services (WSUS) server within your organization.

Analyzing remote systems requires that the RPC services on the remote system are accessible from your MBSA console. This can pose some challenges when working across networks or local firewalls. RPC services can be limited to using a particular range of ports which may ease the burden for firewall administrators. MBSA has limited reporting capabilities. The MBSA console allows you to print or copy to clipboard individual system reports. The MBSA command line tool will allow you to output XML results if you have WSUS within your environment. A PowerShell solution to output the results of multiple scans to a single comma separated variable file can be found in the appendix.

2.1.4. Tenable Network Security Nessus

Nessus is a well-known vulnerability scanner. It is easily deployed and has an extensive set of plug-ins to detect thousands of operating system and application vulnerabilities. Each plug-in is well documented and includes industry references to obtain more information. Nessus is frequently used as a network vulnerability scanner to scan networks for active hosts, discover listening services, and probe services for vulnerabilities. This is effective even without the advantage of credentials. With SMB or Secure Shell (SSH) credentials it becomes an effective vulnerability, configuration, and patch assessment tool for Windows and UNIX based systems. If already using it for network vulnerability scanning it may not need any additional firewall allowances to support the patch assessment capabilities. Nessus needs to be able to access the remote registry service and can optionally start and stop this service if it is not already enabled on target hosts.

Creation of a Nessus scan job involves identifying target systems and choosing a scan policy. Nessus has a policy wizard with a specific template geared toward patch audits. This template will assure that any plug-in dependencies are satisfied to ensure reliable results. As plug-ins are executed against a host, they create a knowledgebase of information that is used to execute additional dependent plug-ins. When conducting Windows patch audits with Nessus, look for successful evaluation of the Microsoft Patch Bulletin Feasibility Check plug-in. This plug-in confirms that the credentials are valid and the configuration of the target systems is such that a patch audit is feasible and results will be accurate. The patch audit plug-ins output identifies the file version information for missing patches. Nessus can export scan results in various levels of detail and in multiple formats.

2.2. Analysis Method

The goal of the analysis is to identify any discrepancies in patches that are reported to be missing. If a particular patch is identified as missing by any one of the four tools, it will be included in the results. Our sampling includes 25 Windows servers running Windows 2008 Standard 32-bit, Windows 2008 R2, and Windows 2012 Standard operating systems. Server roles include Active Directory Domain Services, Active

Directory Federation Services, Internet Information Server, Microsoft SQL Server, SharePoint, and various business and infrastructure applications. The scope of patches includes all Microsoft security bulletins for which patches have been released. Common third party applications for the Microsoft Windows operating system, such as Oracle Java and Adobe Reader, generally have a simple versioning scheme without the applicability logic complexities present within operating system components.

Patch vendors release regular updates to the vulnerability and patch metadata. The schedule and distribution methods vary by vendor. To establish a comparable set of metadata, all of the tools have been updated with patch content that includes representation for all Microsoft updates up to and including those released on 14 October 2014. The original release date for the metadata that include security bulletins MS14-056 through MS14-063 of each tool is provided below. See appendix for additional validation measures.

- This information is not readily accessible in Lumension.
- Symantec CCS Patch Assessment Content Update version 2014-21 released 17 October 2014.
- Microsoft wsusscn2.cab file digitally signed 14 October 2014.
- Nessus plugins were published 15 October 2014.

2.3. Results Comparison

Each unique security update that has not been consistently reported as missing across all four of our tools is identified in Table 1 below. Each is identified by the security bulletin ID and the relevant knowledge base article number. The Microsoft security bulletin will be the authoritative reference from which we will analyze the affected host. If the tool reporting the missing result provides a reason why, it will be compared to the bulletin data. Remember, not all of the tools will report a reason for the patch to be missing. Based on our comparison results and the host analysis, a determination will be made whether to deploy the patch or document a reporting exception – as distinct from exceptions that may exist for application compatibility reasons. If the decision is to deploy, the result is shown. In either case, possible causes for the discrepancy will be evaluated. Keep in mind that the objective is not to evaluate the tools to determine which is better, but to evaluate the discrepancy and seek resolution.

In the context of the organization, Lumension Patch and Remediation is the deployment tool. All Microsoft security updates are deployed monthly. As such, installation failures aside, any unexpectedly missing updates represent a determination by Lumension that an update is not needed. Symantec Control Compliance Suite is the reporting tool. Microsoft Baseline Security Analyzer and Nessus are not generally part of the patch management program but serve the analysis with additional perspectives.

Table 1

Bulletin ID	KB	Product Name	CCS	LEMSS	MBSA	Nessus	Machine Name
MS08-069	954430	MSXML 4.0	☒^{1}			☒^{1}	SVR09
MS08-069	954430	MSXML 4.0	☒^{1}			☒^{1}	SVR05
MS08-069	954430	MSXML 4.0	☒^{1}			☒^{1}	SVR05B
MS09-043	947319	Microsoft Office 2003 Web Components	☒	☒^{2}	☒		SVR01
MS12-034	2676562	Windows Server 2008 R2 Standard (x64)	☒		☒		SVR17B
MS12-057	2687501	Access Database Engine 2010	☒	☒^{2}	☒		SVR11
MS12-070	2716427	SQL Server 2005 x64 Standard Edition	☒				SVR01
MS12-070	2716427	SQL Server 2005 x64 Standard Edition	☒				SVR02B
MS12-070	2716429	SQL Server 2005 x64 Standard Edition	☒				SVR01
MS12-070	2716429	SQL Server 2005 x64 Standard Edition	☒				SVR02B
MS14-001	2837577	Microsoft Word Server 2010 x64	☒	☒^{2}			SVR10B
MS14-016	2923392	Windows Server 2008 R2 Standard (x64)	☒				SVR07B
☒: Missing / ☒^{1}: Missing, unsupported / ☒^{2}: Missing, superseded							

2.3.1. Vulnerabilities in Microsoft XML Core Services Could Allow Remote Code Execution (955218)

Windows Server 2008 R2 Standard (x64)		
Product Support End Date: 04/12/2014		
Bulletin ID	KB Article	Superseded by
MS08-069	954430	None
CCS:	Missing	MSXML4.DLL 4.10.9404.0 < 4.20.9870.0
Lumension:	Software	Update not categorized as security or service pack.
MBSA:	--	
Nessus:	Unsupported	
Action taken:	Deploy SP3 and MS13-002, except Server 2012, rename DLL.	
Action result:	Success. MS XML 4 SP3 is no longer supported.	

The systems which CCS reported this patch to be missing are Windows 2012 servers. Other systems have the msxml4.dll but were reported by CCS as missing an MS XML 4 service pack. Lumension categorizes this update as a "software" package not an update to MS XML 4. MBSA gives no indication that there is a missing update or service pack. The Lumension software package for MS XML 4 service pack 3 was successfully deployed. It does not address the issue with lack of support.

A blog maintained by Alton Blom indicates that MS XML 4 is not a component of a particular operating system release and hence did not follow a typical support lifecycle. He was later told that MS XML 4 was considered a "tool" for support purposes (Blom, n.d.). It seems MS XML 4 has perplexed more than a few Windows administrators. If it does not come with the Windows operating system then where does it come from? Based on the prevalence of MS XML 4 in the organizations environment, on even the latest Windows desktop and server builds, it was clearly part of one of the standard endpoint products. After confirming it was not present following the default Windows installation, standard endpoint applications were installed one at a time. We monitored for the presence of the msxml4.dll. The DLL was present following the installation of the LEMSS Patch and Remediation Module. Version 8 of the Patch and Remediation Module no longer installs MS XML 4.

2.3.2. Vulnerabilities in Microsoft Office Web Components Could Allow Remote Code Execution (957638)

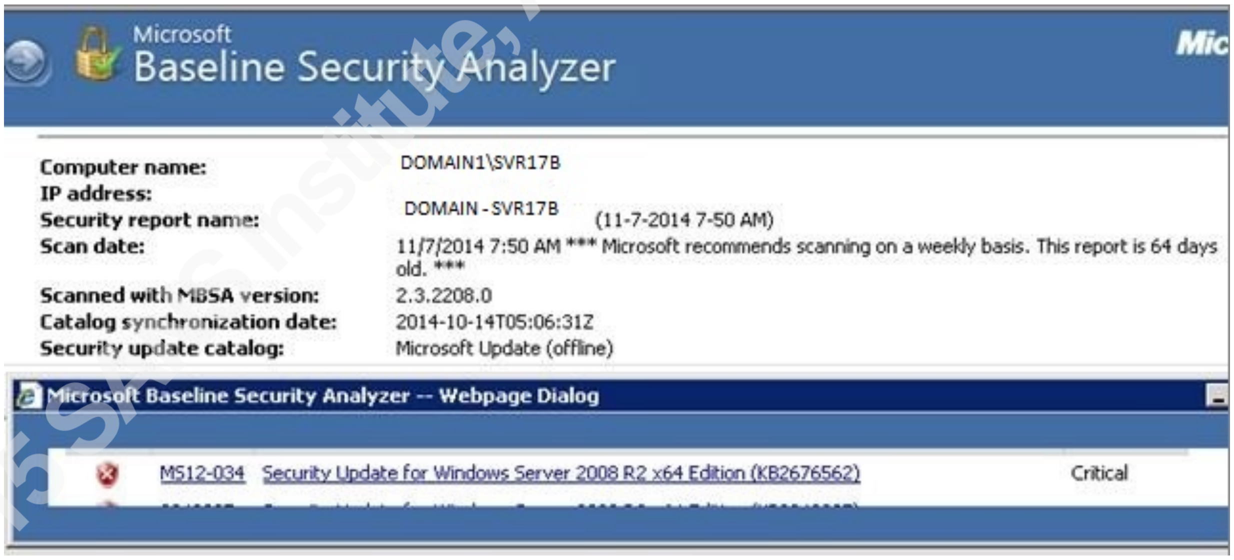
Microsoft Office 2003 Web Components Service Pack 3		
Extended Support End Date: 4/8/2014		
Bulletin ID	KB Article	Superseded by
MS09-043	947319	None
CCS:	Missing	OWC11.DLL 11.0.8166.0 < 11.0.8304.0
Lumension:	Missing (5)	(5) Update is marked superseded
MBSA:	Missing	
Nessus:	--	
Action taken:	Deployed superseded update via Lumension.	
Action result:	Success. File version updated to 11.0.8304.0.	

Ultimately, the problem here is that support for the product has ended. This may be the reason that Lumension has indicated the update is superseded. Lumension support does not recommend deploying superseded updates. Despite a successful deployment of the superseded Lumension update package, the next required action here is to either remove or upgrade the affected software. There is no direct indication provided by any of the tools that the product is no longer supported.

2.3.3. Combined Security Update for Microsoft Office, Windows, .NET Framework, and Silverlight (2681578)

Windows Server 2008 R2 Standard (x64)		
Product Support End Date: 01/14/2020		
Bulletin ID	KB Article	Superseded by
MS12-034	2676562	None
CCS:	Missing	Unknown
Lumension:		
MBSA:	Missing	
Nessus:	--	
Action taken:	Re-deploy patch via Lumension.	
Action result:	Success.	

Both CCS and MBSA reported this patch to be missing. Strangely, the file version information was reported by CCS to be "UNKNOWN". After enabling a Lumension deployment option to install even if the update had been previously installed, the update installed successfully. Figures 1a through 1c show the MBSA result prior to deployment, the deployment date of the hotfix, and the MBSA result post deployment.


Microsoft Baseline Security Analyzer

Computer name: DOMAIN1\SVR17B
IP address:
Security report name: DOMAIN - SVR17B (11-7-2014 7-50 AM)
Scan date: 11/7/2014 7:50 AM *** Microsoft recommends scanning on a weekly basis. This report is 64 days old. ***
Scanned with MBSA version: 2.3.2208.0
Catalog synchronization date: 2014-10-14T05:06:31Z
Security update catalog: Microsoft Update (offline)

Microsoft Baseline Security Analyzer -- Webpage Dialog

MS12-034 Security Update for Windows Server 2008 R2 x64 Edition (KB2676562) Critical

Figure 1a

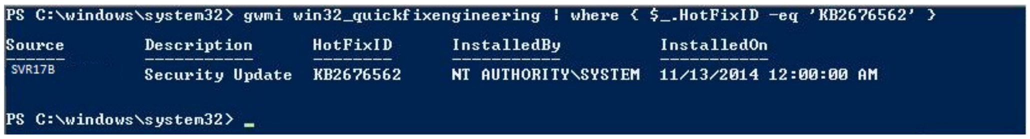
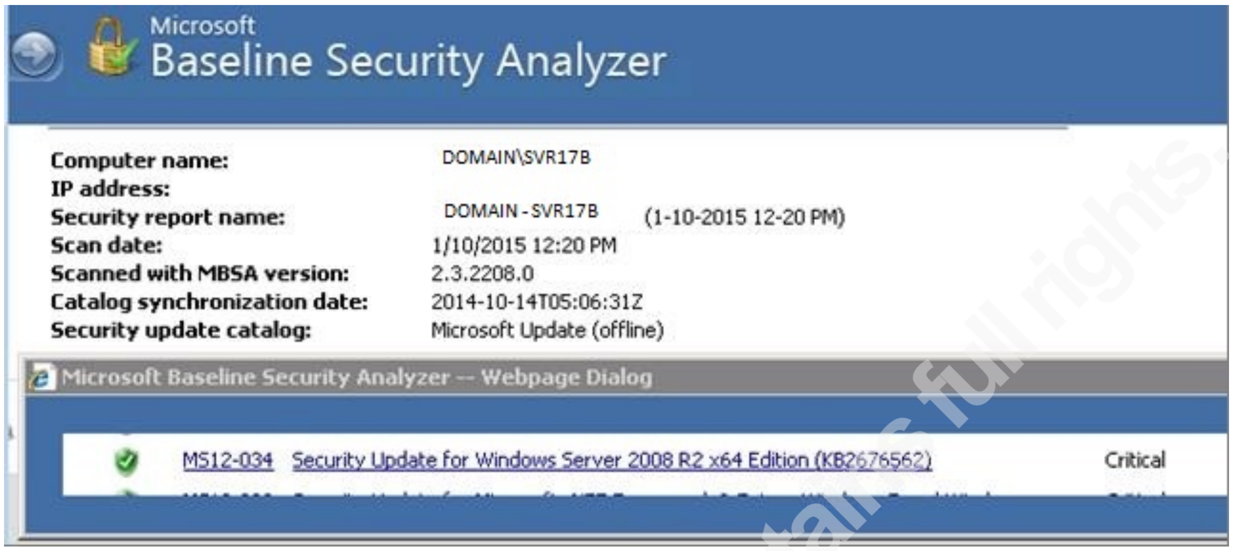

```
PS C:\windows\system32> gwmi win32_quickfixengineering | where { $_.HotFixID -eq 'KB2676562' }

Source        Description      HotFixID      InstalledBy          InstalledOn
------        -----------      --------      -----------          -----------
SVR17B        Security Update  KB2676562     NT AUTHORITY\SYSTEM  11/13/2014 12:00:00 AM

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

Figure 1b

Jason Simsay, jason.simsay@gmail.com



Computer name: DOMAIN\SVR17B
IP address:
Security report name: DOMAIN - SVR17B (1-10-2015 12-20 PM)
Scan date: 1/10/2015 12:20 PM
Scanned with MBSA version: 2.3.2208.0
Catalog synchronization date: 2014-10-14T05:06:31Z
Security update catalog: Microsoft Update (offline)

Microsoft Baseline Security Analyzer -- Webpage Dialog

MS12-034 Security Update for Windows Server 2008 R2 x64 Edition (KB2676562) Critical

Figure 1c

2.3.4. Vulnerability in Microsoft Office Could Allow Remote Code Execution (2731879)

Microsoft Office 2010 Service Pack 1		
Service Pack Support End Date: 10/14/2014		
Bulletin ID	KB Article	Superseded by
MS12-057	2687501	None
CCS:	Missing	MSO.DLL 14.0.4760.1000 < 14.0.6129.5000
Lumension:	Missing (5)	(5) Update is marked superseded
MBSA:	Missing	
Nessus:	--	
Action taken:	Deployed superseded update via Lumension.	
Action result:	Success. File version updated to 14.0.6129.5000.	

In this case, support for the service pack has ended. Again, this could be the reason that Lumension has flagged the update as superseded. The superseded update package was successfully installed. The next required action here is to install Service Pack 2. There is no direct indication from any of the tools that the service pack has reached the end of support, albeit very recently.

2.3.5. Vulnerability in SQL Server Could Allow Elevation of Privilege (2754849)

Microsoft SQL Server 2005 for x64-based Systems Service Pack 4		
Service Pack Support End Date: 04/12/2016		
Bulletin ID	KB Article	Superseded by
MS12-070	2716429	None
CCS:	Missing	SQLSERVR.EXE 2005.90.5057.0 < 2005.90.5069.0
Lumension:	--	
MBSA:	--	
Nessus:	--	
Action taken:	Request exception: Reporting Services is not installed.	
Action result:	Exception approved.	

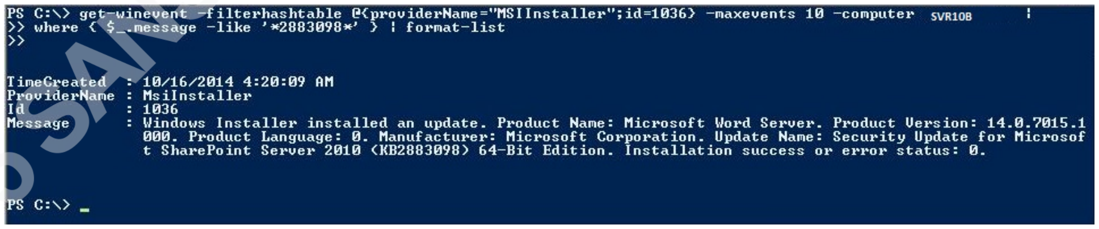
This security bulletin was discussed earlier, in the Predictions section, as an example of an update that has some additional applicability logic. Examining the Nessus Attack Scripting Language (NASL) code for the relevant plug-in confirms the existence of this additional logic. The evaluations vary somewhat for the different SQL versions but the presence of a registry key that indicates reporting services installed is common to all. Reporting services is not installed on the hosts and the update is not offered by Microsoft or Lumension. CCS reports that both the GDR (General Distribution Release) and QFE (Quick Fix Engineering) hot-fixes are missing, refer to Table 1, which seems anomalous as only one or the other can be applicable. Despite Microsoft's indication that public exploit code is likely and Nessus reports that public exploits are available, there is no module available for Rapid7's Metasploit and nothing listed in Offensive Security's Exploit DB. An exception was approved based on the current SQL server configuration lacking the vulnerable reporting services component and also the apparent anomaly with regard to both the QFE and GDR patches missing.

Jason Simsay, jason.simsay@gmail.com

2.3.6. Vulnerabilities in Microsoft Word and Office Web Apps Could Allow Remote Code Execution (2916605)

Microsoft Sharepoint Server 2010 Service Pack 1		
Service Pack Support End Date: 10/14/2014		
Bulletin ID	KB Article	Superseded by
MS14-001	2837577	2878220 in MS14-017
CCS:	Missing	B6040E579AF3CAD43A018FEBA6B225AF
Lumension:	Missing (5)	(5) Update is marked superseded
MBSA:	--	
Nessus:	--	
Action taken:	Deployed superseded update via Lumension.	
Action result:	Success. Registry key now present.	

Checking the Bulletin Search spreadsheet, this update was superseded by 2878220 in MS14-017 which was superseded by 2883098 in MS14-061. MS14-061 was released on 14 October 2014 which coincides with the end of support for Service Pack 1. It is time to get Service Pack 2 deployed. There is no direct indication from any of the tools that the Service Pack is out of support. It is interesting that CCS is looking for a registry key that is not identified in the security bulletin. The superseded Lumension update did populate the registry key. MBSA did not report this update to be missing. The CCS analysis was conducted on 24 October 2014. Consulting the Lumension admin console indicates that MS14-061, which twice supersedes MS14-001, was deployed on 16 October 2014. The event log shown in Figure 2 confirms the patch installation date.



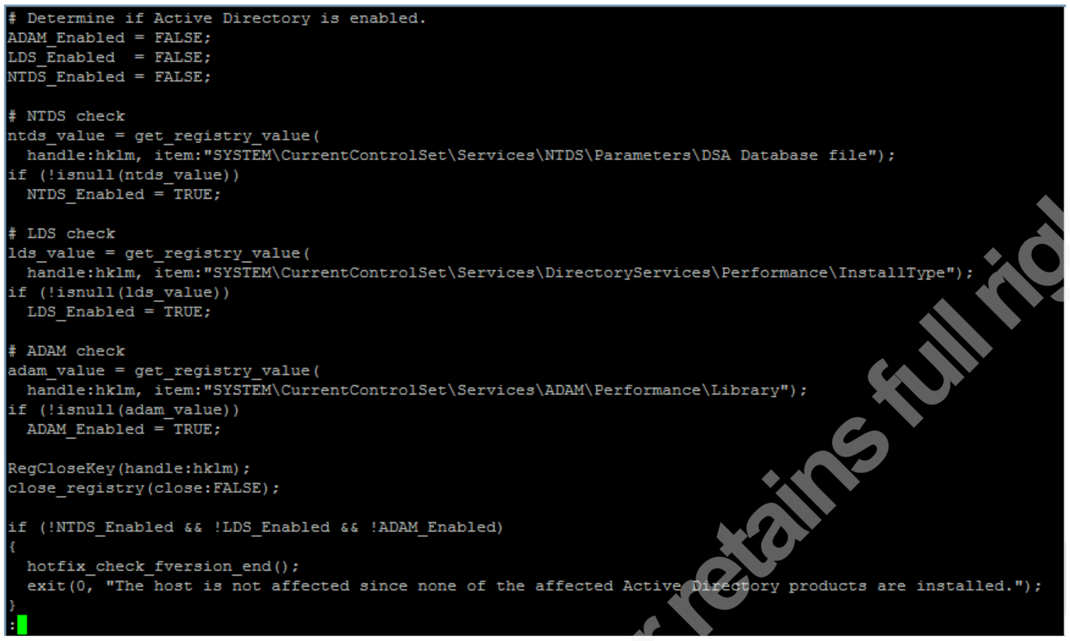
```
PS C:\> get-winevent -filterhashtable @{providerName="MSIInstaller";id=1036} -maxevents 10 -computer SVR10B    |
>> where { $_.message -like '*2883098*' } | format-list
>>


TimeCreated  : 10/16/2014 4:20:09 AM
ProviderName : MsiInstaller
Id           : 1036
Message      : Windows Installer installed an update. Product Name: Microsoft Word Server. Product Version: 14.0.7015.1
               000. Product Language: 0. Manufacturer: Microsoft Corporation. Update Name: Security Update for Microsof
               t SharePoint Server 2010 (KB2883098) 64-Bit Edition. Installation success or error status: 0.



PS C:\> _
```

Figure 2

2.3.7. Vulnerability in Security Account Manager Remote (SAMR) Protocol Could Allow Security Feature Bypass (2934418)

Windows Server 2008 R2 Standard (x64)		
Product Support End Date: 01/14/2020		
Bulletin ID	KB Article	Superseded by
MS14-016	2923392	None
CCS:	Missing	SAMLIB.DLL 6.1.7600.16385 < 6.1.7601.22579
Lumension:	--	
MBSA:	--	
Nessus:	--	
Action taken:	Request exception: AD DS, AD LDS, ADAM not installed.	
Action result:	Exception approved.	

The security bulletin indicates that the update is only offered to Windows 2008 R2 systems that are running Active Directory (AD) Domain Services. Examining the NASL code shown below in Figure 3 confirms the existence of prerequisite logic to check the registry for a value that would indicate AD Domain Services, AD Lightweight Directory Services, or AD Application Mode is installed. These roles are not present on the host and the update is not offered via Microsoft Update. If the prerequisite registry key is created then Nessus will report this update to be missing. This discrepancy and the preceding one suggest that the Shavlik engine used by CCS patch assessment identifies the unpatched DLL, and reports the host to be missing the patch, regardless of a current configuration that is not vulnerable.



```
# Determine if Active Directory is enabled.
ADAM_Enabled = FALSE;
LDS_Enabled  = FALSE;
NTDS_Enabled = FALSE;

# NTDS check
ntds_value = get_registry_value(
  handle:hklm, item:"SYSTEM\CurrentControlSet\Services\NTDS\Parameters\DSA Database file");
if (!isnull(ntds_value))
  NTDS_Enabled = TRUE;

# LDS check
lds_value = get_registry_value(
  handle:hklm, item:"SYSTEM\CurrentControlSet\Services\DirectoryServices\Performance\InstallType");
if (!isnull(lds_value))
  LDS_Enabled = TRUE;

# ADAM check
adam_value = get_registry_value(
  handle:hklm, item:"SYSTEM\CurrentControlSet\Services\ADAM\Performance\Library");
if (!isnull(adam_value))
  ADAM_Enabled = TRUE;

RegCloseKey(handle:hklm);
close_registry(close:FALSE);

if (!NTDS_Enabled && !LDS_Enabled && !ADAM_Enabled)
{
  hotfix_check_fversion_end();
  exit(0, "The host is not affected since none of the affected Active Directory products are installed.");
}
:
```

Figure 3

3. Lessons Learned

3.1. Microsoft Product and Service Pack Lifecycle

The *Ten Principles of Microsoft Patch Management* is a document that warrants considerable understanding: sooner rather than later. The first and second principles suggest that you make Service Packs the "foundation" of your patch management program and that you make the product support lifecycle a "key element" in that program (Budd, 2006). Strangely, if your patch reporting tools are Microsoft based, and had you not discovered these principles, it could take a while to catch on. This is especially true if the priority of your organizations patch management program is to deploy security updates. Operating system Service Packs are high profile, same for SQL server, maybe Office, and .NET, but for Visual C++, Report Viewer, and XML; the list becomes more and more obscure. The latter are not products that are installed out of a direct need but are typically accepted for installation as prerequisites to more apparent software packages. This advocates strongly for effective inventory collection and then consulting the product lifecycle search page and identifying the end of extended support dates. The lesson to be learned is that patch tools are not enough. Organizational processes must address software products and Service Pack life cycles.

3.2. Service Packs Again

The comparison includes three missing patches relevant to the Microsoft Office product line where support for the product or Service Pack has now ended. This includes SharePoint. In reviewing the missing update from the MS14-001 bulletin, a string of updates superseded the update from MS14-001. MS14-061 was released in October and had been applied in a timely manner to the SharePoint 2010 Service Pack 1 system. Coinciding with that 14 October release was the end of support for Service Pack 1. Microsoft's December 2014 security bulletins include MS14-081 which supersedes MS14-061. MS14-081 does not apply to Service Pack 1 but you can be sure the vulnerability is present. From this point forward the vulnerabilities present on systems with Office 2010 without Service Pack 2 will not be patched. The Office product support lifecycle and policy differs from the operating systems as it is based on years and not on versions (Microsoft [2]). Based on the prevalence in the analysis, it is worth keeping a close eye on this distinction.

3.3. Use Multiple Tools

Nicastro tells us that organizations should not use just one tool (Nicastro, p. 139). In the general context that is presented, this is a certainty, but it may be a bit of an understatement as even two tools would benefit from a third. Manzuik has identified patch management as an integral part of the broader discipline of vulnerability management (Manzuik, p. 40). If patch management processes are already leveraging two tools, one for deployment and another for reporting, now is a good time to add credentialed patch audit capabilities to network vulnerability assessment processes. It is imperative that these processes continually assess the need to install not only new updates but older ones as well. As system configurations change and develop over time, updates that didn't apply in the past could apply now. Avoid the inclination to discount the odd tool out that reports a patch as missing and investigate it.

3.4. Maintain Gold Images

It is advantageous to be able to deploy new systems, especially workstations, into the environment in short order. Typically, an organization will drop a gold or base image onto a physical or virtual system, deploy patches, and have it up and running in short order. Likely, this gold image will have the latest OS Service Pack but does it have all the other Service Packs? If not, are those Service Packs deployed prior to putting it into service? If they are, it could take multiple recursive patch deployments to bring the system into compliance with current patches. If not, the installed software could already be unsupported. The team that is responsible for routine patch deployments is going to struggle if new systems requiring multiple patch deployments are continually introduced to the environment.

3.5. Patch/Vulnerability Metadata Update Cycles

Patch management vendors are dependent on the software vendor release cycles. Once the software vendor releases vulnerability data the patch management vendors can package the logic to deliver to customers. They may deliver the vulnerability logic separate from the update packages to speed the process along. The software vendor might not release on time or they may re-release (Microsoft, p. 20). This will have an impact on the patch management vendors. Once the vulnerability metadata is updated there may be some additional lag time before the vulnerable systems can be identified. Do patch agents need to check in to get vulnerability definitions and then later report back the results after a detection scan? Unless you are patching manually, you are not going to be deploying patches to your test systems in the moments after Microsoft releases them. Furthermore, trying to use your patch tool to deploy updates prior to reaching metadata equilibrium with Microsoft can be challenging. Your patch reporting tool might report unexpectedly missing patches following your test roll out. Do you have time to deploy to test systems and assess again before your production roll out window?

4. Conclusion

Having considered the review of basics of patch management, what could the future hold for improvements in patch detection methodologies? Why do we have discrepancies and how do we reduce them? If all patch management tools would report the same set of missing patches then we could be more certain we are fully patched. Computer systems would be more secure.

It seems clear that the trouble lies in the variability of the logic to determine applicability. At times it is straight forward but there are occasions where the logic has more than the usual number of operands. Add to those occasions different update binaries for different versions, different life-cycles, re-releases and superseded updates, it becomes difficult to track. A standard for patch applicability detection and for installation validation is in order. Is there hope?

Patent application US 2007/0169079 A1 has been submitted by Bryan R. Keller describing a system that would collect update metadata that was “formatted or tagged (e.g., using XML) according to some common schema or format” (Keller, p. 2). The patent application considers “prerequisite rules, applicability rules, and installed rules” (Keller, p 4). If the common schema included definition of “installed” rules it would close the vulnerability analysis loop.

The Industry Consortium for Advancement of Security on the Internet (ICASI) has defined an XML specification for the Common Vulnerability Reporting Framework (CVRF). A collaborative effort of leading security vendors and stakeholders established a common format for reporting vulnerability data with the release of CVRF 1.0 in May 2011 (ICASI, n.d.). Many software vendors, as well as MITRE’s Common Vulnerabilities and Exposures site have adopted the CVRF and provide downloads of vulnerability information in the standard format. An April 2012 white paper discusses the enhancements included in the May 2012 release of CVRF 1.1 and also a look ahead to the future of the standard. Subsequent releases may include “Producer Specific Tags” that would enable software vendors to add functionality by including their own XML tags (Schiffman, p. 21). This would seem an excellent possibility to deliver enhanced applicability and installed logic. Ideally, any information contained within a vendor security release would be included and then could be digested by a patch management tool to create vulnerability metadata. Patch management tools could more readily provide that last bit of insight: the patch is not currently applicable but might become applicable if the host configuration state changes or the patch is not installed but due to current configuration the host is not vulnerable. This would make the process more certain, more efficient, and more secure.

References

Blom, Alton. (2014, July 15). And it's official; MSXML 4.0 SP3 is out of support! [Web blog comment]. Retrieved from https://altonblom.com/s34e09/

Budd, Christopher. Ten principles of Microsoft patch management. (2006, May 4). Retrieved from http://technet.microsoft.com/en-us/library/cc512589.aspx

Cole, E., Fossen, J., Northcutt, S., & Pomeranz, H. (2005). SANS Security essentials. (Vol. 5).

Explanation of how patch scanning detection works with Shavlik Protect. (2013, January 1). Retrieved from https://community.shavlik.com/docs/DOC-2259

Gula, R. (2009, February 20). Misleading patch audits. Retrieved from http://www.tenable.com/blog/misleading-patch-audits [1]

Gula, R. (2009, July 27). When patch auditing tools collide. Retrieved from http://www.tenable.com/blog/when-patch-auditing-tools-collide [2]

Internet Consortium for Advancement of Security on the Internet (ICASI). (n.d.) The common vulnerability reporting framework (CVRF). Retrieved from http://www.icasi.org/cvrf

Keller, B., Tariq, S., & Bell, C. (2007, July 19). US 2007/0169079 A1. Location: Google patents.

Manzuik, S., Gold, A., & Gatford, C. Network security assessment: From vulnerability to patch. Rockland, MA: Syngress Publishing, Inc., 2007.

Microsoft Corporation. (2011). Microsoft security update guide (2nd Ed.). Retrieved from http://go.microsoft.com/?linkid=9712840 [1]

Microsoft Office Products Support Lifecycle FAQ. (2013, April 1). Retrieved from http://support2.microsoft.com/gp/lifeoffice [2]

Nicastro, Felicia M. Security patch management (2nd Ed.). Boca Raton, FL: CRC Press, 2012.

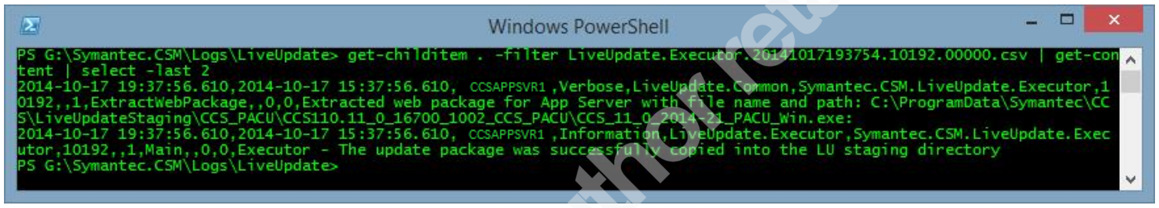
Schiffman, Mike. (2012, April). The missing manual: CVRF 1.1. Retrieved from http://www.icasi.org/docs/ICASI_CVRF1.1_White_Paper.pdf

Symantec to acquire Bindview. (2005, Oct 3). Retrieved from http://www.symantec.com/about/news/release/article.jsp?prid=20051003_02

Jason Simsay, jason.simsay@gmail.com

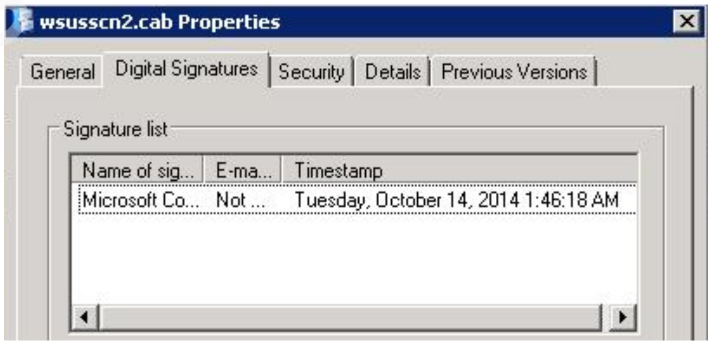
Appendix
Patch Metadata Validation

Symantec released Patch Assessment Content Update (PACU) 2014-21 on 17 October 2014. The release notes that are linked from the following URL indicate that the update includes content representing updates MS14-056 through MS14-063. http://www.symantec.com/security_response/securityupdates/detail.jsp?fid=ccs&pvid=pac&year=&suid=20141017_00. The excerpt from the CCS LiveUpdate log file shown here reflects the PACU in process of being deployed to the environment.



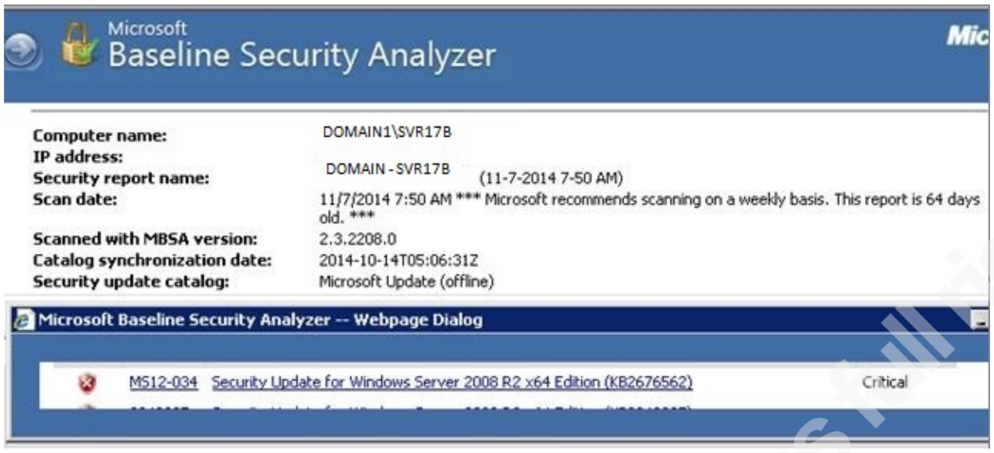
```
PS G:\Symantec.CSM\Logs\LiveUpdate> get-childitem . -filter LiveUpdate.Executor.20141017193754.10192.00000.csv | get-con
tent | select -last 2
2014-10-17 19:37:56.610,2014-10-17 15:37:56.610, CCSAPPSVR1 ,Verbose,LiveUpdate.Common,Symantec.CSM.LiveUpdate.Executor,1
0192,,1,ExtractWebPackage,,0,0,Extracted web package for App Server with file name and path: C:\ProgramData\Symantec\CC
S\LiveUpdateStaging\CCS_PACU\CCS110.11_0_16700_1002_CCS_PACU\CCS_11_0_2014-21_PACU_Win.exe:
2014-10-17 19:37:56.610,2014-10-17 15:37:56.610, CCSAPPSVR1 ,Information,LiveUpdate.Executor,Symantec.CSM.LiveUpdate.Exec
utor,10192,,1,Main,,0,0,Executor - The update package was successfully copied into the LU staging directory
PS G:\Symantec.CSM\Logs\LiveUpdate>
```

Microsoft updated the wsusscn2.cab file commensurate with the Patch Tuesday security bulletin releases on 14 October 2014. The digital signature timestamp present in the properties of the file reflects this date. The MBSA scan result figure shown earlier also reflects the synchronization date for the CAB file that was present during the scan.



Name of sig...	E-ma...	Timestamp
Microsoft Co...	Not ...	Tuesday, October 14, 2014 1:46:18 AM

Jason Simsay, jason.simsay@gmail.com

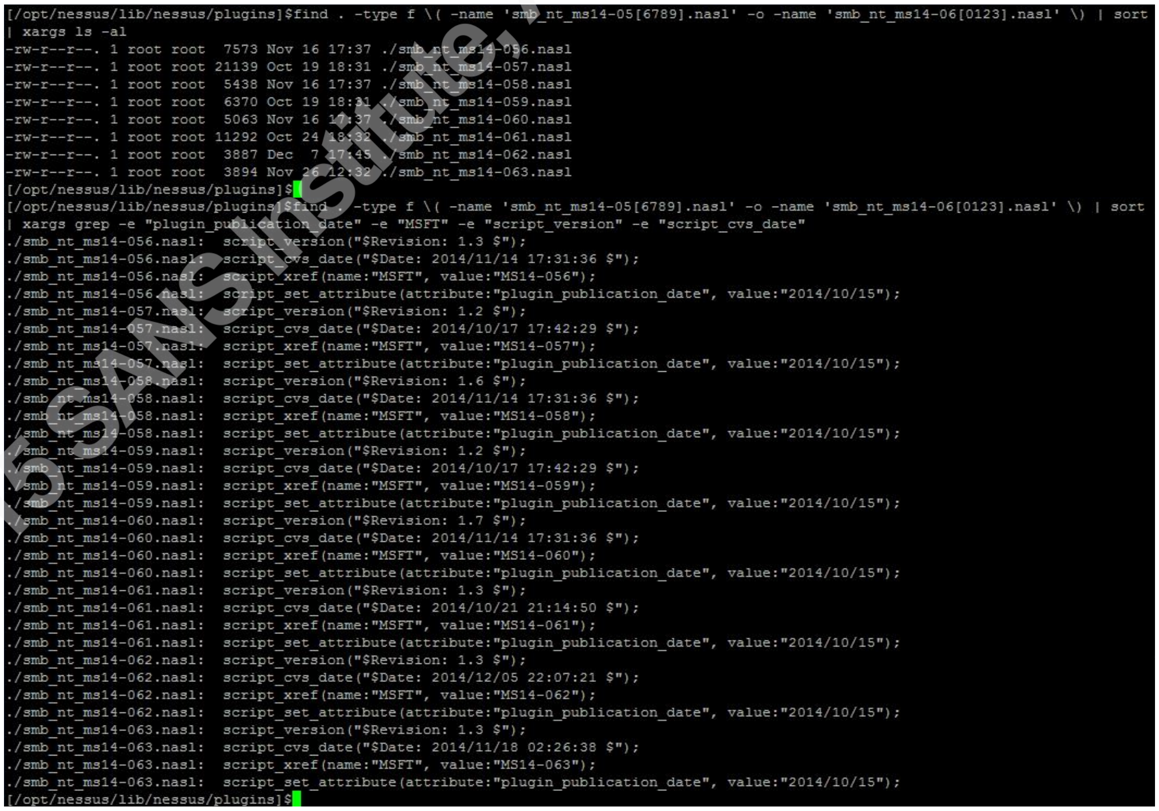


Computer name: DOMAIN1\SVR17B
IP address:
Security report name: DOMAIN - SVR17B (11-7-2014 7-50 AM)
Scan date: 11/7/2014 7:50 AM *** Microsoft recommends scanning on a weekly basis. This report is 64 days old. ***
Scanned with MBSA version: 2.3.2208.0
Catalog synchronization date: 2014-10-14T05:06:31Z
Security update catalog: Microsoft Update (offline)

Microsoft Baseline Security Analyzer -- Webpage Dialog

MS12-034 Security Update for Windows Server 2008 R2 x64 Edition (KB2676562) Critical

The Nessus plugin publication date was originally 15 October 2014. This is reflected in the output of the second Find command below which shows the lines from the NASL script matching “plugin_publication_date”. All have had subsequent modifications based on the “revision”. The “script_cvs_date” is essentially the script modification date and this loosely correlates to the NASL file timestamps in the first Find command output.



```
[/opt/nessus/lib/nessus/plugins]$find . -type f \( -name 'smb_nt_ms14-05[6789].nasl' -o -name 'smb_nt_ms14-06[0123].nasl' \) | sort
| xargs ls -al
-rw-r--r--. 1 root root  7573 Nov 16 17:37 ./smb_nt_ms14-056.nasl
-rw-r--r--. 1 root root 21139 Oct 19 18:31 ./smb_nt_ms14-057.nasl
-rw-r--r--. 1 root root  5438 Nov 16 17:37 ./smb_nt_ms14-058.nasl
-rw-r--r--. 1 root root  6370 Oct 19 18:31 ./smb_nt_ms14-059.nasl
-rw-r--r--. 1 root root  5063 Nov 16 17:37 ./smb_nt_ms14-060.nasl
-rw-r--r--. 1 root root 11292 Oct 24 18:32 ./smb_nt_ms14-061.nasl
-rw-r--r--. 1 root root  3887 Dec  7 17:45 ./smb_nt_ms14-062.nasl
-rw-r--r--. 1 root root  3894 Nov 26 12:32 ./smb_nt_ms14-063.nasl
[/opt/nessus/lib/nessus/plugins]$
[/opt/nessus/lib/nessus/plugins]$find . -type f \( -name 'smb_nt_ms14-05[6789].nasl' -o -name 'smb_nt_ms14-06[0123].nasl' \) | sort
| xargs grep -e "plugin_publication_date" -e "MSFT" -e "script_version" -e "script_cvs_date"
./smb_nt_ms14-056.nasl:  script_version("$Revision: 1.3 $");
./smb_nt_ms14-056.nasl:  script_cvs_date("$Date: 2014/11/14 17:31:36 $");
./smb_nt_ms14-056.nasl:  script_xref(name:"MSFT", value:"MS14-056");
./smb_nt_ms14-056.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-057.nasl:  script_version("$Revision: 1.2 $");
./smb_nt_ms14-057.nasl:  script_cvs_date("$Date: 2014/10/17 17:42:29 $");
./smb_nt_ms14-057.nasl:  script_xref(name:"MSFT", value:"MS14-057");
./smb_nt_ms14-057.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-058.nasl:  script_version("$Revision: 1.6 $");
./smb_nt_ms14-058.nasl:  script_cvs_date("$Date: 2014/11/14 17:31:36 $");
./smb_nt_ms14-058.nasl:  script_xref(name:"MSFT", value:"MS14-058");
./smb_nt_ms14-058.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-059.nasl:  script_version("$Revision: 1.2 $");
./smb_nt_ms14-059.nasl:  script_cvs_date("$Date: 2014/10/17 17:42:29 $");
./smb_nt_ms14-059.nasl:  script_xref(name:"MSFT", value:"MS14-059");
./smb_nt_ms14-059.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-060.nasl:  script_version("$Revision: 1.7 $");
./smb_nt_ms14-060.nasl:  script_cvs_date("$Date: 2014/11/14 17:31:36 $");
./smb_nt_ms14-060.nasl:  script_xref(name:"MSFT", value:"MS14-060");
./smb_nt_ms14-060.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-061.nasl:  script_version("$Revision: 1.3 $");
./smb_nt_ms14-061.nasl:  script_cvs_date("$Date: 2014/10/21 21:14:50 $");
./smb_nt_ms14-061.nasl:  script_xref(name:"MSFT", value:"MS14-061");
./smb_nt_ms14-061.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-062.nasl:  script_version("$Revision: 1.3 $");
./smb_nt_ms14-062.nasl:  script_cvs_date("$Date: 2014/12/05 22:07:21 $");
./smb_nt_ms14-062.nasl:  script_xref(name:"MSFT", value:"MS14-062");
./smb_nt_ms14-062.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
./smb_nt_ms14-063.nasl:  script_version("$Revision: 1.3 $");
./smb_nt_ms14-063.nasl:  script_cvs_date("$Date: 2014/11/18 02:26:38 $");
./smb_nt_ms14-063.nasl:  script_xref(name:"MSFT", value:"MS14-063");
./smb_nt_ms14-063.nasl:  script_set_attribute(attribute:"plugin_publication_date", value:"2014/10/15");
[/opt/nessus/lib/nessus/plugins]$
```

Jason Simsay, jason.simsay@gmail.com

Appendix
outputMBSAScanToCSV.ps1

```
#-------------------------------------------------------------------------------
#Modified: 01/12/2015
#Created by:  Jason Simsay
#Original File Name:  outputMBSAScanToCSV.ps1
#Disclaimer:  This script is provided as is and any use is at your own risk.
#       You will need to provide consent by entering "YES" when prompted.
#Description:  This Powershell script will read properly formatted hostnames
#       (i.e DOMAIN\HOSTNAME) from a text file and search for complete Microsoft
#       Baseline Security Analyzer scans on a specified date.  Discovered scans
#       will be extracted and output to a CSV file.  The MBSA command line
#       interface executable is used to discover and extract scan results.
#Prerequisites:
#       1. Microsoft Baseline Security Analyzer 2.x is installed in
#               "C:\Program Files\Microsoft Baseline Security Analyzer 2"
#       2. MBSA scans have been run and are present on local system
#Script inputs:
#       1. targets.txt, could also be used as input to MBSACLI.EXE
#       2. User supplied date in MM-DD-YYYY format
#       3. User is prompted to continue based on discovered scans
#Script outputs:
#       1. Scan discovery status results are output to STDOUT (screen).
#               Scans that are "Incomplete" are excluded from discovery.
#       2. Scan results for "Missing" or "Installed" patches are output
#               to output.csv
#       3. Error.log will capture STDERR, but will typically be deleted.
#               Change the $ErrorActionPreference="Continue" to retain error.log
#-------------------------------------------------------------------------------

# Obtain user consent that use of this script is at your own risk.
$consent = read-host "This script is provided as is and can be used at your own risk.
Enter 'YES' to proceed"
switch ($consent) {
        "YES" {}
        default {exit}
}
$ErrorActionPreference="SilentlyContinue"

# targets.txt should contain a single FQDN (i.e. DOMAIN\HOSTNAME) per line.
# The format is the same as that which MBSACLI accepts for the /listfile parameter.
# Example command line syntax for scan:
# mbsacli /listfile "targets.txt" /nvc /n "OS+SQL+IIS+Password"
$targets = gc targets.txt

# The $command variable should point to the location of mbsacli.exe.  It is called
# from invoke-expression to be able to squelch the typical output from mbsacli.exe.
$command = "& 'C:\Program Files\Microsoft Baseline Security Analyzer 2\mbsacli.exe'"

$date = read-host "Enter date to retrieve scan results in mm-dd-yyyy format"
#$date="12-11-2014"

# Reformat date to drop leading zeros on month or day part
$date -match '^(?<month>\d{1,2})-(?<day>\d{1,2})-(?<year>\d{4})' | out-null
$dateSearchString = ($matches.month -replace '^0','')+'-'+($matches.day -replace
'^0','')+'-'+($matches.year)

# Declare arrays
$validMatchingScans=@()
$unmatchedHosts=@()
$patchResults = @()

# Scan discovery loop
$targets | foreach {
        $hostName = $_

        # Execute mbsacli.exe with /l to list all scans
        $matchingScans = (invoke-expression -command "$command /l 2>error.log" |
        # Match lines containing hostname
        select-string (($_ -replace '\\','\\') + ',') |
        # Match lines containing date
        select-string $dateSearchString |
        # Do not match lines containing "incomplete" scans
        select-string "Incomplete Scan" -notmatch |
        # Select only 1 scan if more than one successful scan exists for a given
        # host on the specified date.  Potentially troublesome if the most recent
        # scan is more desirable, as the output from mbsacli is a text based sort.
        select -first 1)
        # If no matching scans discovered, add unmatched host to array, otherwise
        # add matching scan to array.
        switch ($matchingScans -eq $null) {
             $true { $unmatchedHosts += $hostname }
             $false { $validMatchingScans += $matchingScans }
        }
}

# Warn if valid scans were not found for a given host
if ($unmatchedHosts.count -gt 0) {
        write-host -foregroundcolor Yellow "WARNING: Could not find valid scans for the
following"$unmatchedHosts.count" hosts."
        $unmatchedHosts
}

# Display the matching scans that were discovered
write-host "The following scan results have been selected for output to CSV:"
$validMatchingScans

# Solicit user for input to continue with CSV output based on
# unmatched hosts and discovered scans
$continue = read-host "Continue with output to CSV (Y/N)"

switch ($continue) {
        "Y" {

                # Walk through discovered scans
                $validMatchingScans | foreach {
                        $scanName = ($_ -split ',' | select -last 1) -replace '^\s',''
                        $hostName = ($_ -split ',' | select -first 1)

                        # Execute mbsacli with /ld switch to display detailed report
                        invoke-expression -command "$command /ld `$scanName 2>>error.log" |

                        # Select only those results where patch is missing or installed
                        select-string -caseSensitive "Missing","Installed" | foreach {

                                # Cleanup whitespace and split on pipe
                                $data = ((($_ -replace '\s{2,}','') -replace '\|\s','|') -
replace '\s\|','|') -split '\|'

                                # Parse fields based on position
                                "{0} {1} {2} {3} {4}" -f
$data[1],$data[2],$data[3],$data[4],$data[5] | out-null

                                # Extract KB number from description field
                                $data[3] -match '\((?<kbnum>KB\d*)\)' | out-null
                                $kb = $matches.kbnum

                                # Add fields to hash table
                                $hash = @{
                                        computer = $hostname
                                        bulletin = $data[1]
                                        kb = $kb
                                        status = $data[2]
                                        description = $data[3]
                                        criticality = $data[4]
                                }

                                # Create new custom object with properties of hash
```

```
                              $obj = new-object psCustomObject -property $hash

                              # Add custom object to results array

                              $patchResults += $obj

                              # Clear values from loop variables
                              clear-variable kb,matches
                      }
               }

               # Select elements from array and export to csv
               $patchresults | select computer,bulletin,kb,status,criticality,description
| export-csv output.csv -notypeinformation
       }

       # Do nothing if user response to continue is other than 'Y'
       default {}
}

# Check size of error.log file to see if output was captured
if ((get-childitem error.log).length -eq 0) {

       # If not delete, else warn.
       remove-item error.log }
       else { write-host -foregroundcolor Red "Errors reported.  Check error.log file in
current directory" }
```