# MONITOREO DE VULNERABILIDADES EN SERVICIOS DE RED PARA EMPRESAS CON NAGIOS IMPLEMENTADO

CAMILO ALBERTO MÉNDEZ LEÓN

UNIVERSIDAD PILOTO DE COLOMBIA
DIRECCIÓN DE POSTGRADOS
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTÁ D.C.
2015

# MONITOREO DE VULNERABILIDADES EN SERVICIOS DE RED PARA EMPRESAS CON NAGIOS IMPLEMENTADO

CAMILO ALBERTO MÉNDEZ LEÓN

Trabajo de grado para optar al título de Especialista en Seguridad Informática.

Tutor:
Ing. Álvaro Escobar

UNIVERSIDAD PILOTO DE COLOMBIA
DIRECCIÓN DE POSTGRADOS
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTÁ D.C.
2015

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

Firma del presidente del jurado

_____

Firma del jurado

_____

Firma del jurado

## DEDICATORIA

Este trabajo va dedicado principalmente a mi esposa Alexandra Rosero, quien fue la persona que siempre estuvo impulsándome a trabajar continuamente en este proyecto hasta finalmente culminarlo. También por la paciencia, y todo el tiempo que me ha permitido “robarle” y que hemos tenido que sacrificar como familia y pareja, con el objetivo de ser mejores profesionales y tener un futuro mejor.

Al igual que a mi padre Alberto Méndez, quien a pesar de haber estudiado unos pocos años, siempre me recuerda que el estudio es la mejor herencia que se le puede dejar a un hijo, y que es algo que como seres humanos no nos podrán arrebatar jamás.

## AGRADECIMIENTOS

Al Ing. Álvaro Escobar, actual director de la especialización en seguridad informática de la Universidad Piloto de Colombia, por sus acertadas orientaciones respecto a las dudas que tuve durante el desarrollo del trabajo, al igual que por la gran confianza que me mostró durante las respectivas tutorías.

Igualmente, agradezco a mi esposa, ingeniera industrial, por los aportes realizados basada en sus conocimientos y experiencia, los cuales están reflejados en la cuantificación de la eficiencia y eficacia de los escáneres de vulnerabilidades evaluados.

**CONTENIDO**

**pág.**

## LISTA DE FIGURAS

**pág.**

## LISTA DE TABLAS

**pág.**

## LISTA DE ANEXOS

**pág.**

# 0. INTRODUCCIÓN

La seguridad informática es un concepto que poco a poco está tomando fuerza e importancia en las empresas del país. Esto se debe básicamente a 2 situaciones: la primera situación es al alto índice de robo de datos y ataques a la información que se conocen en el actual mundo digital y que son ampliamente difundidos por los medios de comunicación como la televisión, periódicos, etc. La segunda situación es el continuo trabajo que han venido realizando los profesionales en esta área en cuanto a concienciar acerca de la importancia y sensibilidad que tiene la protección de la información y el impacto reputacional que puede acarrear consigo.

Muchos empresarios en Colombia conocen la importancia de la seguridad informática pero hasta la fecha son muy pocos los que hacen algo por implementarla al interior de sus compañías. Razones para ello normalmente son la desinformación de la gerencia y los costos o niveles de retorno muy bajos con respecto a los beneficios que esta pueda traer. Sin embargo, la mayoría de estos empresarios le presta una gran atención e importancia al monitoreo de las redes, ya que habitualmente son conscientes de la gravedad de un incidente que afecte la red, pues este podría afectar en la mayoría de casos la producción de la empresa, lo que en definitiva se traduce en pérdidas para el negocio.

En el mercado existen muchos sistemas capaces de hacer monitoreo de redes, pero para una empresa con recursos limitados, dicha variedad se limita a algunos pocos sistemas. Adicional a ello, cuando en el mercado existe un producto con las características siguientes, la elección se facilita:

- Es una de las herramientas mayormente usada por la industria para el monitoreo de redes.
- Posee una gran variedad de documentación gratuita proporcionada tanto por el desarrollador como por la comunidad.
- Es software libre.

El sistema que reúne las características anteriores se conoce como Nagios. Este es un sistema que en la actualidad es común encontrarlo en las empresas a nivel mundial debido a sus capacidades, flexibilidad y costo nulo en su principal versión.

Normalmente es usado para monitorear hardware, software, servicios de red, conmutadores, enrutadores entre otros, pero en muy pocos casos para monitorear temas relativos a la seguridad como las vulnerabilidades de los sistemas.

Nagios es un sistema que se difunde por medio de Internet, y por esta razón la documentación más relevante tanto oficial como no oficial viene por este mismo medio. En las investigaciones previas realizadas al proyecto, se evidencia que existe poca información y ninguna herramienta que le permita a Nagios llevar un monitoreo profundo del estado de vulnerabilidad de un sistema.

En este proyecto se creará un procedimiento que sirva como referente de implementación de un sistema de monitoreo de vulnerabilidades basado en Nagios que permita aprovechar algunas de las capacidades que se ausentan en algunas herramientas de seguridad como lo son:

- Notificación de eventos a los contactos definidos vía correo o SMS entre otros.
- Provee una vista centralizada de estado completo de los sistemas monitoreados.
- Reportes de notificaciones, alertas y SLAs.
- Acceso vía web – multiusuario.
- Rápida detección de incidentes.

# 1. PLANTEAMIENTO DEL PROBLEMA

## 1.1 DESCRIPCIÓN Y FORMULACIÓN DEL PROBLEMA

Pensando en todas aquellas empresas que cuentan con una infraestructura tecnológica crítica para el negocio la cual deba mantenerse protegida tanto a nivel de disponibilidad como de seguridad, y que por temas relacionados con la falta de recursos tanto financieros como técnicos, no es posible la puesta en marcha y posterior mantenimiento de un escáner de vulnerabilidades efectivo, se plantea el siguiente interrogante:

¿Qué se requiere para usar el sistema Nagios para monitorear el estado de las vulnerabilidades de un servicio determinado?

En el desarrollo de este proyecto se estudiarán y se probarán las herramientas de software libre que mejor posibilidades tengan en ser integradas con Nagios, eligiendo al final del ejercicio la mejor opción buscando como parámetro principal la interoperabilidad con el sistema Nagios.

## 1.2 JUSTIFICACIÓN

De alcanzarse el objetivo se espera que al final del ejercicio se obtenga un procedimiento de implementación que sirva de referencia a todos los administradores de red o agentes de seguridad que requieran de un sistema como el siguiente:

- Capaz de monitorear y centralizar la información referente a la disponibilidad y vulnerabilidad de los servicios.
- Envío de notificaciones de eventos, al correo o vía SMS, no solo referentes al hardware, software y servicios de red, sino también a las vulnerabilidades de los sistemas monitoreados.
- Visualizar gráficamente y en tiempo real mediante la WEB el estado de las vulnerabilidades de seguridad de los sistemas monitoreados.
- Obtener una única herramienta de monitoreo para los equipos de TI y Seguridad de la información. Esto a su vez implica una reducción de costos sustancial en cuanto a la implementación del sistema como al mantenimiento.

Con ello se consigue transformar en cierta medida a Nagios, hoy visto como herramienta de monitoreo de redes, en un instrumento de seguridad que permita identificar en tiempo real las deficiencias de seguridad de un sistema, habilitando a los administradores de TI a proveer una pronta solución a las debilidades que se detecten.

## 1.3 OBJETIVOS

**1.3.1 Objetivo general.** Emplear el sistema de monitoreo de redes Nagios para monitorear las vulnerabilidades de un sistema determinado, y a su vez crear el procedimiento que sirva como referente de implementación de Nagios como herramienta para el monitoreo de vulnerabilidades de un sistema.

**1.3.2 Objetivos específicos.** Para conseguir exitosamente el objetivo general de este proyecto, se deben cumplir antes los siguientes objetivos:

- Conocer a fondo el funcionamiento del sistema de monitoreo de redes Nagios.
- Encontrar herramientas de código abierto que se ejecuten bajo Linux y que permitan realizar análisis de vulnerabilidades de servicios.
- Elegir las herramientas que posean mayor flexibilidad para ser integradas en el sistema Nagios y realizar los desarrollos que se requieran para ello.
- Crear un ambiente de pruebas y evaluar las distintas herramientas halladas anteriormente.
- Realizar pruebas de las herramientas sobre un host objetivo Windows que no esté completamente actualizado, de manera que sean evidentes algunas vulnerabilidades.
- Elegir y justificar la mejor herramienta y método para el monitoreo de vulnerabilidades de un sistema determinado.
- Generar el procedimiento de implementación del sistema objeto de éste proyecto.

## 1.4 ALCANCES Y LIMITACIONES DEL PROYECTO

Debido a la amplia variedad de sistemas operativos que encontramos en el mercado y a la experiencia del autor, se ha decidido enfocar el estudio en herramientas de código abierto, y especialmente que se ejecuten en entornos Linux.

Siempre que sea posible, se utilizarán las versiones más recientes tanto de los sistemas operativos como de las herramientas usadas. Sin embargo, por el tiempo que tome el desarrollo del proyecto, no es posible garantizar que al finalizarlo dichas versiones continúen siendo las últimas disponibles.

# 2. MARCO TEÓRICO

## 2.1 EL SISTEMA DE MONITOREO DE REDES NAGIOS CORE

Nagios Core es un sistema de código abierto y una aplicación de monitoreo de redes. Este vigila los host y servicios que se especifiquen, alertando cuando algo malo ocurre o cuando mejora.

Nagios Core fue diseñado originalmente para ejecutarse en Linux, aunque debería funcionar en la mayoría de otros sistemas Unix también.

Algunas de las muchas características de Nagios Core son:

- Seguimiento de los servicios de red (SMTP, POP3, HTTP, NNTP, PING, etc.).
- El monitoreo de los recursos del host (carga del procesador, uso de disco, etc.).
- Diseño simple de plugins que permite a los usuarios desarrollar fácilmente sus propias comprobaciones de servicio.
- Capacidad para definir la jerarquía de red utilizando hosts "padre", lo que permite la detección y la distinción entre hosts que están abajo y los que son inalcanzables.
- Notificaciones a contactos cuando se producen problemas de servicio o de host y estos se resuelven (por correo electrónico, buscapersonas o método definido por el usuario).
- Capacidad para definir controladores de eventos, que se ejecutan durante un evento en los servicios o hosts para la resolución de problemas de manera proactiva.
- Rotación automática del archivo de registro.
- Interfaz web opcional para ver el estado actual de la red, histórico de notificaciones y problemas, el archivo de registro, etc.

**2.1.1 Requisitos del Sistema.** El único requisito de ejecutar Nagios Core es una máquina que ejecute Linux (o variante de UNIX) que tiene acceso a la red y un compilador de C instalado (si se instala desde el código fuente).

No se requiere utilizar el CGI que se incluye con Nagios Core. Sin embargo, si se decide utilizarlos, se tendrá que tener el siguiente software instalado:

- Un servidor WEB (preferiblemente Apache).
- Librerías Thomas Boutell's gd versión 1.6.3 o superior.

**2.1.2 Licenciamiento.** Nagios Core está disponible bajo los términos de la GNU *General Public License* versión 2, publicada por la *Free Software Foundation*. Esto le da permiso legal para copiar, distribuir y/o modificar Nagios bajo ciertas condiciones. Lea el archivo "licencia" en la distribución de Nagios o leer la versión online de la licencia para más detalles.

## 2.2 ESCÁNER DE VULNERABILIDADES DE CÓDIGO ABIERTO

Antes de hablar de los escáneres de vulnerabilidades, es conveniente definir lo que es una vulnerabilidad. Una vulnerabilidad se define en la norma ISO 27002 como "Una de las debilidades de un activo o grupo de activos que pueden ser explotadas por una o más amenazas". De acuerdo a la definición anterior, un escáner de vulnerabilidades puede ser definido como un programa informático que se encarga de identificar las debilidades en las redes, en la infraestructura o en las aplicaciones.

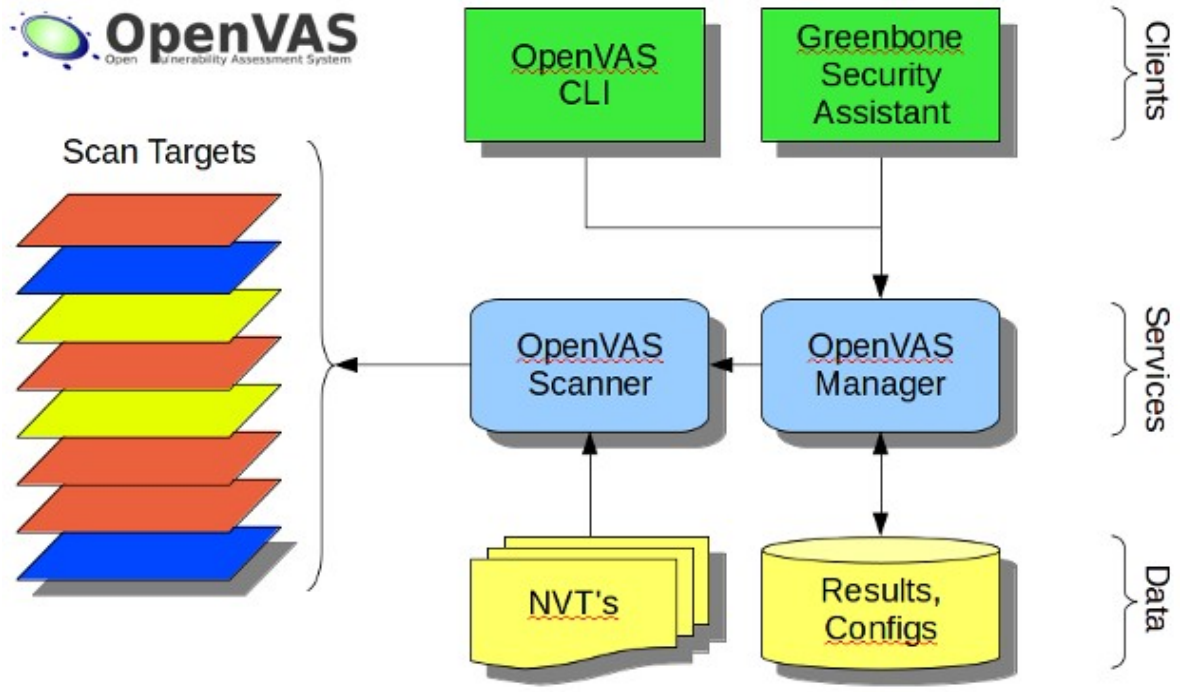
Actualmente en el mercado tenemos una gran variedad de escáner de vulnerabilidades que pueden agruparse en dos grandes grupos: los que requieren de la adquisición de un acuerdo de licencia, y los que son de código abierto o no requiere de pago alguno por su uso. Dentro de la primera categoría podemos encontrar a aplicaciones como NESSUS o SAINT, y en la segunda categoría podemos ubicar a OpenVAS, los cuales son ampliamente conocidos y difundidos en el medio.

En este proyecto nos enfocaremos en los escáneres distribuidos bajo el concepto de código abierto.

**2.2.1 Open Vulnerability Assessment System (OpenVAS).** El sistema de evaluación de vulnerabilidad abierto, denominado inicialmente GnessUs, es un marco de diversos servicios y herramientas que ofrecen una solución completa y potente de escaneo y gestión de vulnerabilidades. Todos los productos que componen OpenVAS son software libre y la mayoría licenciados bajo licencia GNU (GNU GPL).

En la figura 1 se muestra la arquitectura de OpenVAS y como interactúa cada uno de sus componentes con los demás que lo componen.

Figura 1. Arquitectura de OpenVAS.




Fuente: http://www.openvas.org/about.html.

Como se ha podido observar, el núcleo de su arquitectura es el OpenVAS Scanner, el cual ejecuta muy eficientemente las recientes pruebas de vulnerabilidad de red (NVTs) que son servidas a través de actualizaciones diarias mediante el servicio OpenVAS NVT Feed. Sus características son:

- Escaneo concurrente de múltiples nodos.
- Protocolo de transferencia OpenVAS (OTP).
- Soporte SSL para OTP.
- Soporte WMI.

El OpenVAS Manager es el servicio central que consolida el plan de escaneo de vulnerabilidades en una completa solución de gestión de vulnerabilidades. Toda la inteligencia se implementa en el Manager de modo que es posible implementar varios clientes ligeros que se integren consistentemente por ejemplo, en relación con el filtrado o la clasificación de los resultados del análisis. El OpenVAS Manager también controla una base de datos SQL (basada en sqlite) donde se

almacenan de forma centralizada toda la configuración y los datos de resultado de la exploración. Por último, OpenVAS Manager también se encarga de la gestión de los usuarios, incluyendo el control de acceso con los grupos y roles. Sus características son:

- Protocolo de gestión OpenVAS (OMP).
- Base de datos SQL (sqlite) para configuraciones y resultados de escaneos.
- Permite tareas concurrentes de escaneo (varios OpenVAS Scanner).
- Notas de gestión para los resultados de escaneos.
- Gestión de falsos positivos.
- Escaneos programados.
- Reportes en múltiples formatos (XML, HTML, LaTeX, entre otros).
- Modo maestro – esclavo para controlar muchas instancias desde una única central.
- Administración de usuarios.

El asistente de seguridad Greenbone (GSA) es un servicio web ligero que ofrece una interfaz de usuario para los navegadores web. Sus características son:

- Cliente para OMP y el Protocolo de administración de OpenVAS (OAP).
- Soporte de HTTP y HTTPS.
- Servidor web propio (microhttpd), por lo tanto no requiere un servidor web adicional.
- Sistema de ayuda en línea integrada.
- Soporte multi-idioma.

OpenVAS CLI contiene la herramienta de línea de comandos que permite crear procesos por lotes para manejar el OpenVAS Manager. Otra de las herramientas de este paquete es un plugin de Nagios. Sus características son:

- Cliente para OMP.
- Corre en Windows, Linux, etc.
- Incluye plugin para Nagios.

**2.2.2 Security Auditor's Research Assistant (SARA).** Es una herramienta de análisis de red para encontrar vulnerabilidades. SARA reporta los hallazgos por medio de un servidor WEB integrado, ejecutando un análisis de diccionario proporcionado por Common Vulnerabilities and Exposure (CVE). Al usar este diccionario, la herramienta continúa actualizada sin la necesidad de actualizaciones al código o la participación del desarrollador, mejorando las probabilidades de que las vulnerabilidades actuales serán reconocidas por la herramienta siempre y cuando el diccionario se actualice regularmente. Adicional al diccionario CVE, SARA también es compatible con otros tipos de exploraciones.

Además de realizar la evaluación de red, SARA es capaz de realizar un proceso de certificación a nuevos sistemas, como por ejemplo, un servidor WEB que no debe tener más que los servicios de Apache, SSH, respaldo y monitoreo.

SARA funciona en la mayoría de versiones de Unix, Linux, Mac OS X y Windows (usando coLinux), aunque se debe tener en cuenta que su desarrollo fue descontinuado y su última versión disponible es la 7.9.1 de Mayo 1 de 2009.

SARA es un programa Perl que ejecuta tanto el análisis de vulnerabilidades y sirve las páginas web. Se puede ejecutar como una herramienta de línea de comandos o iniciar como un demonio del sistema. Los resultados se presentan a través de un navegador web. SARA también puede usar las características de nmap para identificar los sistemas operativos de los ordenadores escaneados.

Sus características más destacadas son:

- Se integra con la Base de Datos Nacional de Vulnerabilidad (NVD).
- Realiza pruebas de inyección SQL.
- Realiza pruebas exhaustivas XSS.
- Soporte de estándares CVE.
- Modo autónomo o demonio.
- Módulo de búsqueda empresarial.
- Instalación Plug-in para aplicaciones de terceros.
- Uso gratuito abierto licencia SATAN.

**2.2.3 Vulscan.nse, escaneo de vulnerabilidades con nmap.** Vulscan.nse es un módulo que mejora nmap a un escáner de vulnerabilidades. La opción nmap -sV

permite la detección de versiones por servicio que se utiliza para determinar las fallas potenciales de acuerdo con el producto identificado. Los datos son buscados en una versión sin conexión de diferentes bases de datos de vulnerabilidades.

El 06 de marzo de 2010 fue presentada por Marc Ruef, su primera versión con la cual salió al público. La versión inicial está apoyada en la Base de Datos de Vulnerabilidades de Código Abierto (OSVDB por sus siglas en inglés) como base de datos de vulnerabilidad.

El 25 de Junio de 2013 fue puesta a disposición la versión 1.0. En ella se ha realizado una completa re-escritura del script. Ésta mejora mucho la velocidad, la precisión y la confianza de las pruebas. Por otra parte, diferentes bases de datos de vulnerabilidad como scip VulDB, CVE y SecurityFocus ya están disponibles.

El 14 de Agosto de 2013 fue lanzada la versión 2.0 con la opción de Identificación. Este importante lanzamiento introduce identificación de la versión, lo que puede mejorar la exactitud de los resultados, siempre y cuando las bases de datos de vulnerabilidad vinculados proporcionen información sobre la versión (apoyado por la base de datos de scip VulDB sólo en el momento).

# 3. DISEÑO METODOLÓGICO

## 3.1 TIPO DE INVESTIGACIÓN

La investigación tiene un componente descriptivo y otro exploratorio. El componente descriptivo se debe a que el ejercicio busca determinar mediante una serie de pruebas la opción más adecuada para obtener un sistema que permita realizar el monitoreo y seguimiento de vulnerabilidades aprovechando las características de la herramienta Nagios y de los escáneres de vulnerabilidades de código abierto que actualmente se encuentran en el mercado. Sin embargo, existe un componente del trabajo en que se requiere de una investigación de tipo exploratoria, ya que en algún momento el ejercicio se centrará en generar un método de integración entre la herramienta Nagios con los escáneres de vulnerabilidades que se ajusten, tema que a la fecha es poco explorado y los resultados están sujetos a la experimentación del autor.

## 3.2 TÉCNICAS DE RECOLECCIÓN DE DATOS

Las técnicas que recolección de datos que se emplearán, básicamente consiste en la observación de los siguientes parámetros:

- Vulnerabilidades halladas por cada sistema.
- Veracidad de los resultados obtenidos por cada sistema.
- Tiempo requerido por el sistema para completar un análisis de un objetivo.

## 3.3 TÉCNICAS DE ANÁLISIS E INTERPRETACIÓN DE LOS DATOS

Para que los resultados obtenidos sean lo más fiables posibles, es necesario generar unas reglas para que al estudio en cuestión no se le inyecten vicios, provocando que los resultados finales se alteren a favor de un sistema u otro.

Una de las reglas primordiales es que el hardware empleado para todos los sistemas objeto de estudio sea homogéneo, lo cual generará resultados que podrán compararse entre sí, sin la necesidad de tener en cuenta las características de hardware empleadas para cada sistema.

Otra regla es que los sistemas objeto de estudio deberán emplearse en sus versiones más recientes. Aunque no siempre se podrá cumplir debido a temas relativos a compatibilidades de librerías y otras variables no contempladas, este punto es tan importante como el anterior para no alterar los resultados.

Para el análisis de las vulnerabilidades, se emplearán las herramientas que sean necesarias y estén acordes al objetivo, siempre buscando que éstas sean de fácil acceso y de código abierto para no incurrir en costos adicionales.

# 4. DESARROLLO DEL PROYECTO

## 4.1 HOST OBJETIVO PARA PRUEBAS DE LABORATORIO

Para las pruebas que se realizarán a los escáneres de vulnerabilidades, se ha habilitado un host virtual el cual se usará como host objetivo y cuyas características técnicas se describen en la tabla 1.

Tabla 1. Características técnicas del host Windows objetivo.

| Ítem | Detalle |
|---|---|
| Computador: | Virtual – VMWare Workstation 10.0.3 |
| Sistema operativo: | Windows XP |
| Service Pack: | 2 |
| Procesador: | Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. 1 Core asignado |
| Memoria: | 512 MB |
| Disco duro: | 10 GB |
| Dirección IP: | 172.16.16.135 |
| Detalles | Sistema operativo sin actualizaciones, con Apache v1.3.17 instalado con las opciones por defecto y el agente de NRPE Nsclient++ 0.3.9. |

Fuente: autor del proyecto.

## 4.2 PRUEBAS Y DESARROLLOS SOBRE OPENVAS

**4.2.1 Características del servidor OpenVAS.** El servidor de OpenVAS fue implementado junto con el sistema Nagios en una máquina virtual VMWare con las características descritas en la tabla 2:

Tabla 2. Características técnicas del servidor OpenVAS.

| Ítem | Detalle |
|---|---|
| Servidor: | Virtual – VMWare Workstation 10.0.3 |
| Sistema operativo: | Debian Linux 7.6 |
| Versión de Nagios: | Nagios core v3.0.6 |
| Versión de OpenVAS: | OpenVAS Scanner 3.4.0 |
| Procesador: | Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. 1 Core asignado |
| Memoria: | 768 MB |
| Disco duro: | 15 GB |

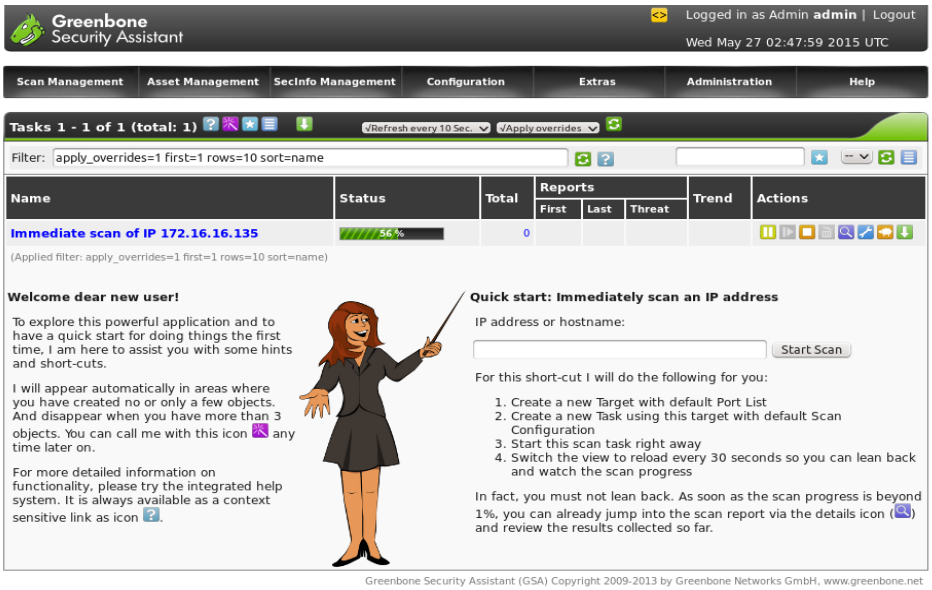
Fuente: autor del proyecto.

**4.2.2 Evaluación inicial.** De acuerdo con los objetivos planteados en el proyecto, es necesario evaluar cada uno de los escáneres de vulnerabilidades de código abierto hallados durante el desarrollo.

La evaluación básicamente consiste en la realización de un escaneo de vulnerabilidades al host objetivo con las herramientas propias del sistema evaluado, con el fin de determinar rápidamente si merece la pena un estudio más profundo y dar continuidad con la siguiente etapa de integración con el sistema Nagios.

Para el caso de OpenVAS, se utiliza el asistente de seguridad Greenbone cuyo acceso se realiza vía web ingresando al puerto 9392 el cual viene definido por omisión. El aspecto del asistente puede verse en la figura 2.

Figura 2. Asistente de seguridad Greenbone de OpenVAS.



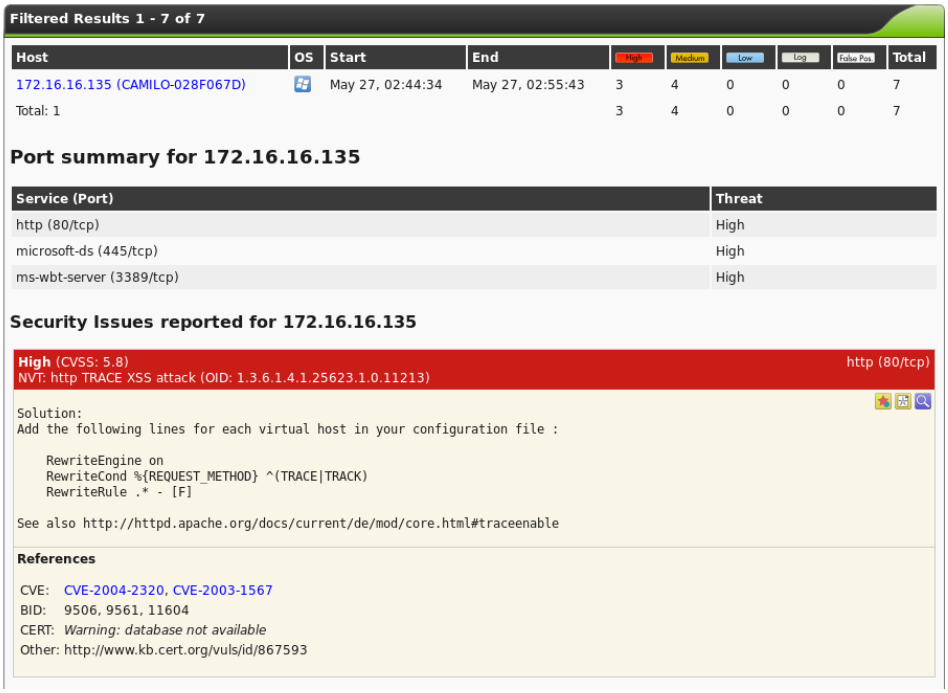

Fuente: autor del proyecto.

El escaneo del host objetivo se realiza mediante la opción *Quick Start: Immediately scan an IP address* la cual se muestra en la pantalla inicial de la herramienta. Para dar inicio, se ingresa la IP del host objetivo y se pulsa el botón *Start Scan*.

Al finalizar el escaneo del host objetivo, se procede a revisar el detalle de las vulnerabilidades halladas las cuales se muestran resumidas en la figura 3.

Para decidir de manera efectiva y rápida su veracidad, se analizaran sólo las vulnerabilidades clasificadas como altas por el sistema. A cada una de estas se le realizará un análisis superficial para determinar si se trata de una vulnerabilidad verdadera del host objetivo.

Figura 3. Detalle de vulnerabilidades en evaluación de OpenVAS.




Fuente: autor del proyecto.

**4.2.2.1 http TRACE XSS attack.** Esta vulnerabilidad se muestra al final de la figura 3 y también es la primera que se encuentra relacionada en el Anexo A. Al realizar el análisis de la información proporcionada por el mismo reporte, como el sitio web al que recomienda visitar y las referencias que se incluyen, se evidencia lo siguiente:

- El software Apache es afectado por dicha vulnerabilidad[1].
- La directiva TraceEnable que permite “deshabilitar” dicha vulnerabilidad fue incluida a partir de las versiones 1.3.34, 2.0.55 y posteriores.
- Hasta el 8 de Enero de 2003 fue notificado Apache de dicha vulnerabilidad, lo que podría entenderse que toda versión lanzada anterior a esa fecha es potencialmente vulnerable. La versión 1.3.17 fue lanzada el 29 de Enero de 2001[2].

1 [Citado en 04 de Junio de 2015] Disponible en < http://www.kb.cert.org/vuls/id/867593 >.

2 [Citado en 04 de Junio de 2015] Disponible en < https://www.apachehaus.com/index.php?option=com_content&view=article&id=119&Itemid=104 >.

Con las razones encontradas podemos pensar que la vulnerabilidad detectada es potencialmente legítima.

**4.2.2.2 Microsoft Windows SMB Server NTLM Multiple Vulnerabilities.** Ésta es la segunda vulnerabilidad detectada y clasificada como alta en el reporte de OpenVAS para la evaluación inicial. La vulnerabilidad hace referencia al Boletín de Seguridad Microsoft MS10-012, el cual afecta al sistema operativo Microsoft Windows XP Service Pack 2[3]. Aclarando que el host objetivo no presenta actualización alguna luego de su instalación, es altamente probable que ésta sea otra vulnerabilidad legítima.

**4.2.2.3 Microsoft RDP Remote Code Execution Vulnerabilities.** Es la tercera y última vulnerabilidad clasificada como alta del reporte en cuestión.

Al validar la referencia CVE-2012-0002[4], se puede comprobar rápidamente que la versión de Microsoft Windows XP Service Pack 2 está afectada por la vulnerabilidad. Como se ha mencionado anteriormente, el host objetivo no ha sido actualizado y por ende se concluye que esta vulnerabilidad puede también ser legítima.

**4.2.3 Conclusión de la evaluación inicial.** Conforme el análisis de las vulnerabilidades altas ha resultado satisfactorio, se procede con los trabajos de integración del sistema Nagios con el escáner de vulnerabilidades OpenVAS.

**4.2.4. Plugin de OpenVAS para Nagios.** Para la integración con el sistema Nagios, fue necesario el desarrollo del plugin nombrado como check_openvas, el cual fue escrito en su totalidad en lenguaje bash script aprovechando la herramienta de línea de comandos OpenVAS CLI basada en comando “omp”.

Cada vez que se ejecuta el plugin con los parámetros adecuados, se realiza el siguiente proceso:

1. Se crea el host objetivo en OpenVAS a partir de la IP, la cual debe ser proporcionada como parámetro. El código que se muestra en la figura 4 se encarga de ello.

3 [Citado en 09 de Junio de 2015] Disponible en < https://technet.microsoft.com/en-us/library/security/ms10-012.aspx >.

4 [Citado en 10 de Junio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-0002 >.

Figura 4. Código para crear objetivo en OpenVAS.

```
# CREAR TARGET
omp -u $USER -w $PASS --xml='
<create_target>
<name>'$TARGET'</name>
<hosts>'$TARGET'</hosts>
</create_target>' > /dev/null

TARGETID=$(omp -u $USER -w $PASS -T | grep "  $TARGET"$ | awk '{ print $1 }')
```

Fuente: adaptada de http://stackoverflow.com/questions/22812048/bash-script-for-openvas-omp.

2. Crea una tarea de escaneo la cual es asociada al host objetivo. El comando usado para ello se muestra en la figura 5, y usa el valor de la variable TARGETID, cuyo valor es generado a partir del código de creación del host objetivo.

Figura 5. Comando para crear tarea de escaneo en OpenVAS.

```
# CREAR TASK
TASKID=$(omp -u $USER -w $PASS --create-task --name=$NAME --config=$TYPE --target=$TA
RGETID)
```

Fuente: adaptada de http://stackoverflow.com/questions/22812048/bash-script-for-openvas-omp.

3. Posteriormente se inicia la tarea de escaneo de vulnerabilidades. Para iniciar el escaneo es necesario indicar el ID de la tarea, el cual fue almacenado en la variable TASKID en el paso anterior. El comando utilizado para ello se muestra en la figura 6.

Figura 6. Comando para crear objetivo en OpenVAS.

```
# START TASK
SCANID=$(omp -u $USER -w $PASS -S $TASKID)
```

Fuente: adaptada de http://stackoverflow.com/questions/22812048/bash-script-for-openvas-omp.

4. Luego es necesario generar un loop con una sentencia *while* que indique el momento en que finaliza el escaneo. Una vez el loop detecta que la tarea ha

finalizado, se genera el reporte en formato de texto y se guarda en un archivo temporal. El código que se encarga de esta tarea se muestra en la figura 7.

Figura 7. Comandos para obtener el reporte del escaneo OpenVAS.

```
while [ -z "$FINISH" ]; do
        sleep 5
        FINISH=$(omp -u $USER -w $PASS -G | grep $TASKID | grep "  Done  ")
done

# GET REPORT
omp -u $USER -w $PASS --get-report $SCANID --format=$FORMAT > $TMPFILE
```

Fuente: adaptada de http://stackoverflow.com/questions/22812048/bash-script-for-openvas-omp.

5. Con el fin de no dejar rastros o configuraciones que posteriormente puedan afectar el rendimiento de OpenVAS, se elimina la tarea y el objetivo creados anteriormente usando el código que aparece en la figura 8.

Figura 8. Comandos para borrar tareas y objetivos creados en OpenVAS.

```
# BORRAR TASK
omp -u $USER -w $PASS -D $TASKID

# BORRAR TARGET
omp -u $USER -w $PASS -X '<delete_target target_id="'$TARGETID'"/>' > /dev/null
```

Fuente: autor del proyecto.

6. Luego, partiendo del reporte generado en el cuarto paso, se captura la información relevante del reporte como lo son la cantidad de vulnerabilidades halladas con sus respectivas clasificaciones, los cuales serán mostrados a la salida de la ejecución del plugin. Estos datos son utilizados para evaluar y emitir los alertamientos *CRITICAL*, *WARNING* ó *OK*. En la figura 9 se muestra el código que realiza esta operación.

Figura 9. Código de extracción de vulnerabilidades del reporte OpenVAS.

```
# READ REPORT
HIGH=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $2 }')
MEDIUM=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $3 }')
LOW=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $4 }')
LOG=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $5 }')
FALSE_POS=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $6 }')
```

Fuente: autor del proyecto.

7. El paso siguiente es evaluar la salida y la alerta que emite el sistema Nagios. El plugin se ha configurado basándose en las siguientes reglas:

• Si el escaneo realizado no evidencia vulnerabilidades altas ni medias, sin importar el resto de vulnerabilidades encontradas como bajas o informativas, la salida del plugin es *OK*.

• Si el escaneo realizado no evidencia vulnerabilidades altas, pero sí evidencia vulnerabilidades medias, sin importar el resto de vulnerabilidades encontradas como bajas o informativas, la salida del plugin es *WARNING*.

• Si el escaneo realizado evidencia vulnerabilidades altas, sin importar el resto de vulnerabilidades encontradas como las medias, bajas o informativas, la salida del plugin es *CRITICAL*.

El código que cumple las reglas mencionadas se muestra en la Figura 10.

Figura 10. Código para generar el tipo de alertamiento al sistema Nagios.

```
if [ $HIGH -eq 0 ] && [ $MEDIUM -eq 0 ]; then
        OUTPUT=OK
        EXIT=0
fi

if [ $HIGH -eq 0 ] && [ $MEDIUM -gt 0 ]; then
        OUTPUT=WARNING
        EXIT=1
fi

if [ $HIGH -gt 0 ]; then
        OUTPUT=CRITICAL
        EXIT=2
fi
```

Fuente: autor del proyecto.

8. El último paso del plugin es generar la salida, cuya función es informar el tipo de alertamiento y los detalles relevantes del reporte como lo son las cantidades de vulnerabilidades encontradas en el escaneo con su respectiva clasificación. Adicional a ello, y tomando en cuenta que el plugin tuvo que ser configurado en modo pasivo, la salida debe generarse en un formato específico, y esta deberá volcarse sobre el archivo nagios.cmd que se encuentra dentro de la jerarquía de directorios de instalación del sistema Nagios. El código que genera esta salida y el volcado de información se muestra en la figura 11.

Figura 11. Código para generar salida del plugin.

```
        TIMESTAMP=$(date +%s)
        SRV_NAME="Vulnerabilities"
        echo [$TIMESTAMP] PROCESS_SERVICE_CHECK_RESULT";"$HOSTNAME";"$SRV_NAME";"$EXI
T";"$OUTPUT: Vulnerabilities: HIGH=$HIGH, MEDIUM=$MEDIUM, LOW=$LOW and LOG=$LOG $(sed
 -n -e '/Port Summary/,/Security Issues/ p' $TMPFILE | grep -v "Security Issues") "|
HIGH=$HIGH MEDIUM=$MEDIUM LOW=$LOW LOG=$LOG" >> /usr/local/nagios/var/rw/nagios.cmd
        exit 0
```

Fuente: autor del proyecto.

**4.2.5 Integración del plugin en Nagios.** Para la integración del plugin, se han realizado las siguientes configuraciones en el sistema Nagios:

• Se ha definido un nuevo servicio asociado al host objetivo tal como se muestra la figura 12:

Figura 12. Configuración del servicio check_openvas.

```
define service{
        use                     generic-service
        host_name               WindowsXP SP2
        service_description     Vulnerabilities
        check_command           check_openvas!admin!openvas
        check_interval          60
        active_checks_enabled   0
        passive_checks_enabled  1
        }
```

Fuente: autor del proyecto.

• Se ha realizado la configuración del comando check_openvas tal como se muestra la figura 13:

Figura 13. Configuración del comando check_openvas.

```
# 'check_openvas' command definition
define command{
        command_name    check_openvas
        command_line    $USER1$/check_openvas $ARG1$ $ARG2$ $HOSTADDRESS$ $HOSTNAME$
        }
```

Fuente: autor del proyecto.

## 4.3 PRUEBAS SOBRE SARA

**4.3.1 Características del servidor SARA.** El servidor de SARA fue implementado junto con el sistema Nagios en una máquina virtual VMWare con las características descritas en la tabla 3:

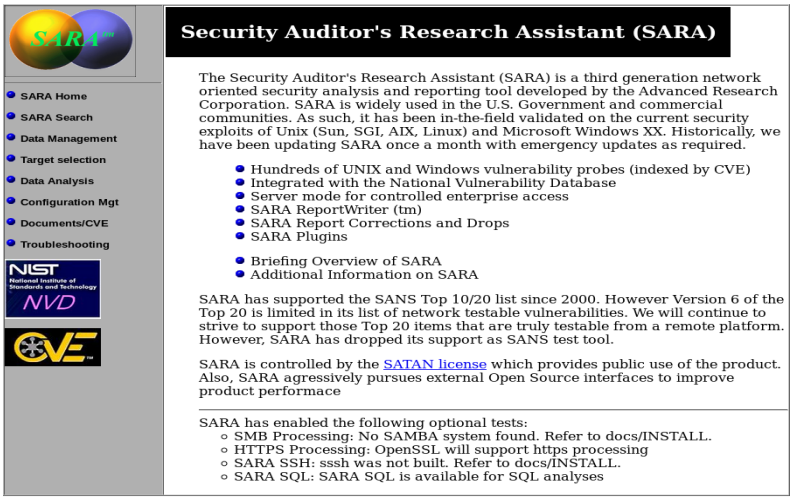
Tabla 3. Características técnicas del servidor SARA.

| Ítem | Detalle |
|---|---|
| Servidor: | Virtual – VMWare Workstation 10.0.3 |
| Sistema operativo: | Debian Linux 7.6 |
| Versión de Nagios: | Nagios core v3.0.6 |
| Versión de SARA: | V7.9.2ª |
| Procesador: | Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. 1 Core asignado |
| Memoria: | 768 MB |
| Disco duro: | 15 GB |

Fuente: autor del proyecto.

**4.3.2 Evaluación inicial.** La evaluación inicial del software se realiza mediante el acceso web al puerto 666, puerto que es utilizado por defecto por la aplicación. En la figura 14 se muestra la pantalla inicial de la consola.

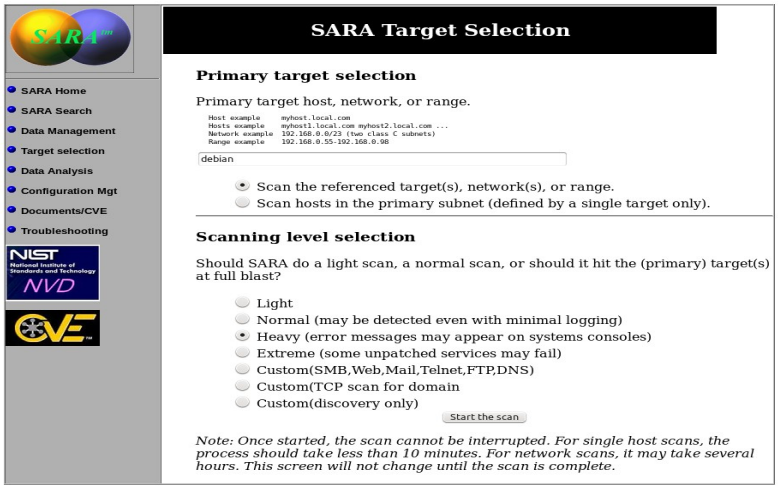
Figura 14. Consola de administración web de SARA.




Fuente: autor del proyecto.

Para realizar el escaneo mediante la herramienta, se debe ingresar por la opción *Target selection* ubicada en el menú izquierdo de la pantalla. Luego de la carga de la pantalla, similar a la que se muestra en la figura 15, se ingresa la IP del host objetivo en el campo marcado para ello y luego se pulsa el botón *Start the scan*.

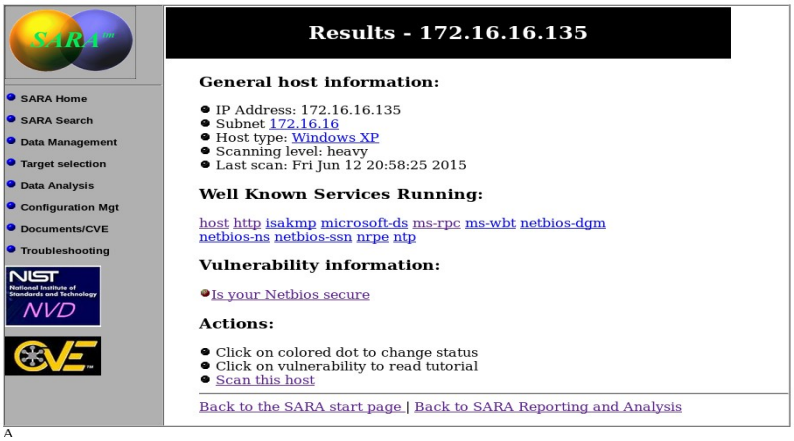
Figura 15. Herramienta para iniciar un escaneo en SARA.




Fuente: autor del proyecto.

Al finalizar el análisis de vulnerabilidades, que tarda no más de un minuto, se evidencia que SARA ha reconocido una vulnerabilidad la cual es identificada con el mensaje *Is your Netbios secure* tal como se muestra en la figura 16.

Figura 16. Reporte de vulnerabilidades de evaluación de SARA.

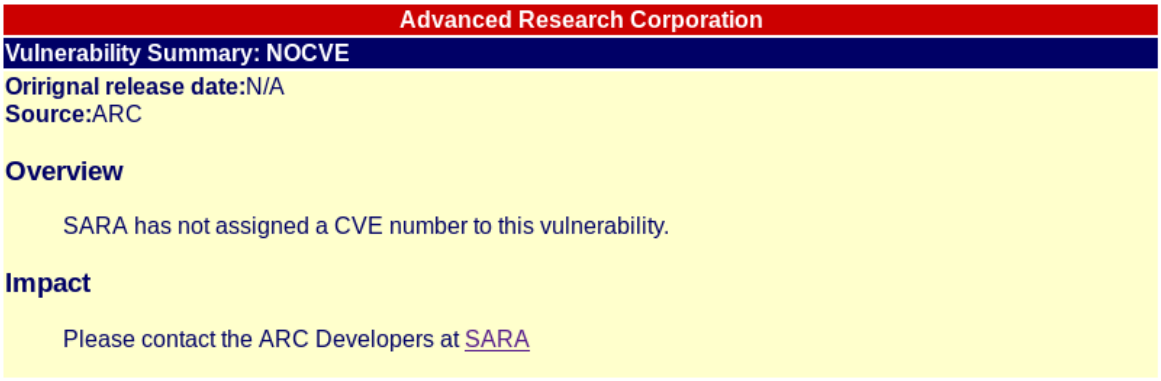


A

Fuente: autor del proyecto.

Al revisar el detalle para dicha vulnerabilidad en la herramienta, no se encuentra información alguna sobre la misma ni tampoco referencias CVE asociadas, tal como se muestra en la figura 17.

Figura 17. Detalle de vulnerabilidad de evaluación de SARA.




Fuente: autor del proyecto.

**4.3.3 Conclusión de la evaluación inicial.** Como ya se tiene un antecedente de vulnerabilidades conocidas proporcionado por OpenVAS, el cual identificó 3

vulnerabilidades altas que potencialmente son legítimas, se puede determinar que el escáner de vulnerabilidades SARA no será objeto de estudio de este trabajo debido a que no ha reconocido ninguna vulnerabilidad legítima sobre el host objetivo que justifique su estudio.

## 4.4 PRUEBAS SOBRE EL MÓDULO NMAP VULSCAN.NSE

**4.4.1 Características del servidor implementado para el módulo Vulscan.nse.** El servidor donde se ha implementado el módulo Vulscan.nse junto con el sistema Nagios, es una máquina virtual VMWare con las características descritas en la tabla 4:

Tabla 4. Características técnicas del servidor para el script Vulscan.nse.

| Ítem | Detalle |
|---|---|
| Servidor: | Virtual – VMWare Workstation 10.0.3 |
| Sistema operativo: | Debian Linux 7.6 |
| Versión de Nagios: | Nagios core v3.0.6 |
| Versión de NMAP: | Nmap v6.00 |
| Versión de Vulscan.nse: | Vulscan.nse v2.0 |
| Procesador: | Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. 1 Core asignado |
| Memoria: | 768 MB |
| Disco duro: | 15 GB |

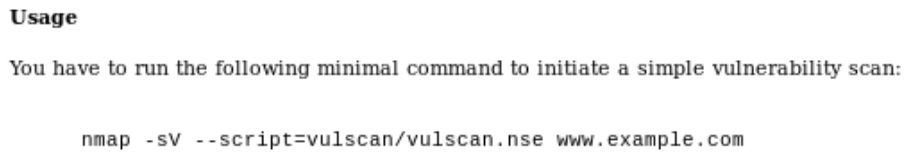
Fuente: autor del proyecto.

**4.4.2 Evaluación inicial.** La evaluación inicial del módulo vulscan.nse es sencilla, ya que por tratarse de un complemento de mnap, no es necesario disponer un entorno gráfico ni sitio web al que debamos ingresar para probarlo. Como nmap puede ser ejecutado utilizando una consola de comandos, la prueba también puede ser realizada por este medio.

De acuerdo a la documentación del desarrollador del complemento[5], el comando reducido para iniciar un análisis de vulnerabilidades simple es el mostrado en la figura 18.

5 [Citado en 20 de Junio de 2015] Disponible en < http://www.computec.ch/projekte/vulscan/?s=documentation >.

Figura 18. Comando reducido para escaneo de vulnerabilidades en vulscan.nse.



**Usage**

You have to run the following minimal command to initiate a simple vulnerability scan:

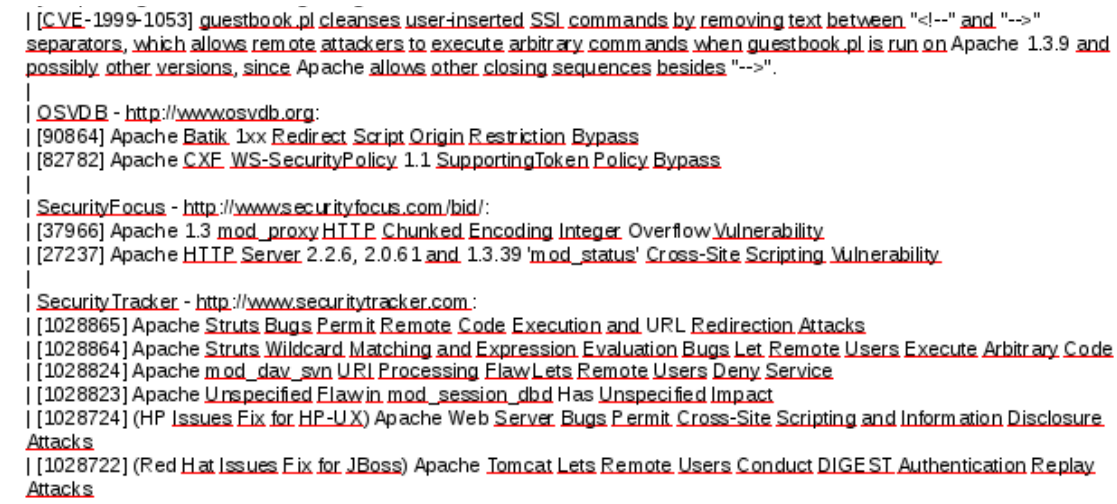
```
nmap -sV --script=vulscan/vulscan.nse www.example.com
```

Fuente: http://www.computec.ch/projekte/vulscan/?s=documentation.

Con ello, se procede entonces a realizar el escaneo de evaluación inicial al host objetivo ejecutando el comando "*nmap -sV --script=vulscan/vulscan.nse 172.16.16.135 > /tmp/Reporte\ Vulscan.txt"* el cual almacena el reporte de vulnerabilidades en el archivo "/tmp/Reporte Vulscan.txt" del sistema operativo donde se ejecuta. El escaneo tomó cerca de un minuto y en el Anexo B se muestra la primera de 511 hojas que arrojó originalmente el reporte.

Al analizar con detenimiento el reporte generado, se puede hacer una aproximación de la manera en que funciona el complemento. En primer lugar, nmap se encarga de encontrar los puertos abiertos, luego determina el servicio que corre allí junto con su respectiva versión, y por último el módulo Vulscan.nse se encarga cruzar los datos obtenidos con cada una de las 8 bases de datos preinstaladas con el mismo complemento. Del cruce de datos de nmap con las bases de datos, las cuales vienen en formato csv (valores separados por coma), se genera una lista de posibles vulnerabilidades que puede afectar a la versión específica del servicio. Lo anterior quiere decir que una misma vulnerabilidad puede ser reconocida hasta 8 veces (1 por cada base de datos), y esto puede ser una explicación al amplio reporte de 511 páginas obtenido al final del ejercicio. Para evitar lo anterior, es posible limitar el complemento Vulscan para que sólo cruce la información con una única base de datos. Derivado de lo anterior, se ha encontrado un inconveniente adicional el cual le resta fiabilidad al reporte. Como se muestra en la figura 19, las bases de datos OSVDB, SecurityFocus y Security Tracker no evidencian las mismas vulnerabilidades. Por ello, se puede concluir que una base de datos puede llegar a ser complemento de otra o varias a la vez.

Figura 19. Extracto del reporte Vulscan.

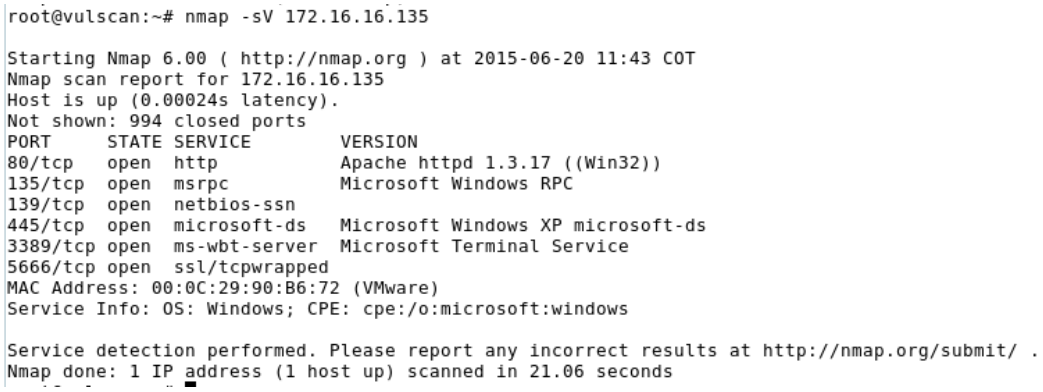



Fuente: autor del proyecto.

Otro inconveniente que puede deducirse rápidamente y que está directamente relacionado con el funcionamiento del complemento, es que la capacidad de detección de vulnerabilidades está directamente relacionada con la capacidad de nmap para detectar las versiones de los servicios. Para observar los servicios a los que nmap ha sido capaz de determinarle la versión, basta con ejecutar el comando reducido para escaneo de vulnerabilidades sin el parámetro que hace referencia al módulo vulscan.nse. El comando en cuestión es *"nmap -sV 172.16.16.135"* cuyo resultado se muestra en la figura 20.

Como puede verse en la imagen, nmap detecta 6 puertos tcp abiertos (80, 135, 139, 445, 3389 y 5666), de los cuales solo a uno de ellos le determina la versión exacta del servicio (Apache httpd 1.3.17). Para 4 de los 6 puertos, le es posible reconocer el nombre del servicio pero no el número de versión (Microsoft Windows RPC, Microsoft Windows XP microsoft-ds, Microsoft Terminal Service y netbios-ssn) y al puerto faltante no le es posible reconocer el nombre del servicio asociado (5666/tcp).

Figura 20. Detección de versiones de servicios de nmap.



```
root@vulscan:~# nmap -sV 172.16.16.135

Starting Nmap 6.00 ( http://nmap.org ) at 2015-06-20 11:43 COT
Nmap scan report for 172.16.16.135
Host is up (0.00024s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE        VERSION
80/tcp   open  http           Apache httpd 1.3.17 ((Win32))
135/tcp  open  msrpc          Microsoft Windows RPC
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds   Microsoft Windows XP microsoft-ds
3389/tcp open  ms-wbt-server  Microsoft Terminal Service
5666/tcp open  ssl/tcpwrapped
MAC Address: 00:0C:29:90:B6:72 (VMware)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.06 seconds
```

Fuente: autor del proyecto.

Otra gran deficiencia encontrada en este complemento, es que el reporte generado no clasifica las vulnerabilidades de acuerdo a su criticidad, lo cual lo hace poco práctico y funcional debido a que pone en un mismo nivel todas las vulnerabilidades detectadas.

Por último, es buena idea pasar a evaluar algunas vulnerabilidades detectadas por el complemento, de modo que se tenga una idea más acertada de la veracidad de sus resultados.

Por la gran cantidad de vulnerabilidades detectadas (aproximadamente 24410) se sospecha que la cantidad de falsos positivos es muy superior a la cantidad de las vulnerabilidades legítimas, y es por ello que la evaluación se centrará en evaluar unas pocas vulnerabilidades, que por su propio nombre, puedan tratarse de falsos positivos.

**4.4.2.1 Microsoft Windows NT 4.0/2000 RPC Server Denial of Service.** Como su nombre lo indica, esta vulnerabilidad afecta a los sistemas operativos Windows NT 4.0 y Windows 2000, información que puede ser confirmada en el sitio web de la base de datos libre scip VulDB[6] y por el Boletín de Seguridad Microsoft MS01-041 al cual hace referencia.

**4.4.2.2 Microsoft Windows NT 4.0/2000 Terminal Server Denial of Service.** Al igual que la vulnerabilidad anterior, ésta afecta a los sistemas operativos Windows NT 4.0 y Windows 2000, información que puede ser confirmada en el sitio web de

6 [Citado en 23 de Junio de 2015] Disponible en < http://www.scip.ch/es/?vuldb.17371 >.

la base de datos libre scip VulDB[7] y por el Boletín de Seguridad Microsoft MS01-052 al cual hace referencia.

**4.4.2.3 Ms ISA Server 2000 Proxy Service Memory Leak Denial of Service.** Como su nombre lo indica, esta vulnerabilidad afecta al software Microsoft ISA Server 2000, información que puede ser confirmada en el sitio web de la base de datos libre scip VulDB[8] y por el Boletín de Seguridad Microsoft MS01-045 al cual hace referencia. Es necesario recordar que el host objetivo no tiene dicho producto instalado, por lo cual no puede estar afectado por tal vulnerabilidad.

**4.4.2.4 CVE-2009-1717.** Esta vulnerabilidad, de acuerdo al *National Institute of Standards and Technology NIST,* afecta a dispositivos Apple Mac OS X 10.5 antes de la versión 10.5.7[9]. Para este caso claramente es un falso positivo ya que el host objetivo tiene como sistema operativo Microsoft Windows XP.

**4.4.2.5 Microsoft Exchange SMTP extended verb request denial of service.** De acuerdo al sitio web de *IBM Internet Security Systems*, esta vulnerabilidad afecta al producto Microsoft Exchange Server 5.5, el cual no está instalado en el host objetivo.

**4.4.3 Conclusión de la evaluación inicial.** Como ya se ha mencionado antes, la cantidad de vulnerabilidades reportadas por este complemento es extremadamente elevado y su evaluación no es sencilla. Sin embargo, mientras se realizaba la elección de las vulnerabilidades a evaluar, se evidenció que muchas de estas vulnerabilidades también podrían ser verdaderas. No obstante, se notó rápidamente que la cantidad de falsos positivos es evidentemente alta, por lo que se decide no continuar con el estudio a fondo del complemento, ya que es una aplicación que necesita aún bastante trabajo para poder compararse con alguno de los escáneres de vulnerabilidades que se encuentran en el mercado actualmente.

7 [Citado en 23 de Junio de 2015] Disponible en < http://www.scip.ch/es/?vuldb.17655 >.

8 [Citado en 23 de Junio de 2015] Disponible en < http://www.scip.ch/es/?vuldb.17375 >.

9 [Citado en 23 de Junio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-1717 >.

# 5. RESULTADOS DEL PROYECTO

## 5.1 DESEMPEÑO DEL PLUGIN CHECK_OPENVAS

Para analizar el desempeño del plugin se tendrá en cuenta los siguientes parámetros a medir, los cuales permitirán cuantificar las características más evidentes del plugin para que puedan ser comparados con otro software de su tipo.

- Funcionalidad y aplicabilidad en un entorno real.
- Tiempo que tarda el plugin en realizar el análisis del host objetivo.
- Cantidad de verdaderos positivos detectados.
- Cantidad de falsos positivos detectados.

**5.1.1 Funcionalidad y aplicabilidad en un entorno real.** Conforme el desarrollo y el estudio realizado en la implementación del plugin check_openvas, se pueden dar por cumplidos los siguientes objetivos:

- Es una solución integrada en Nagios para monitorear vulnerabilidades de un host objetivo.
- Se usa únicamente software de código abierto.
- Muestra en tiempo “real” las vulnerabilidades halladas de un sistema objetivo mediante el entorno web nativo del sistema Nagios.
- La salida del plugin es compatible con el envío de notificaciones del sistema Nagios.

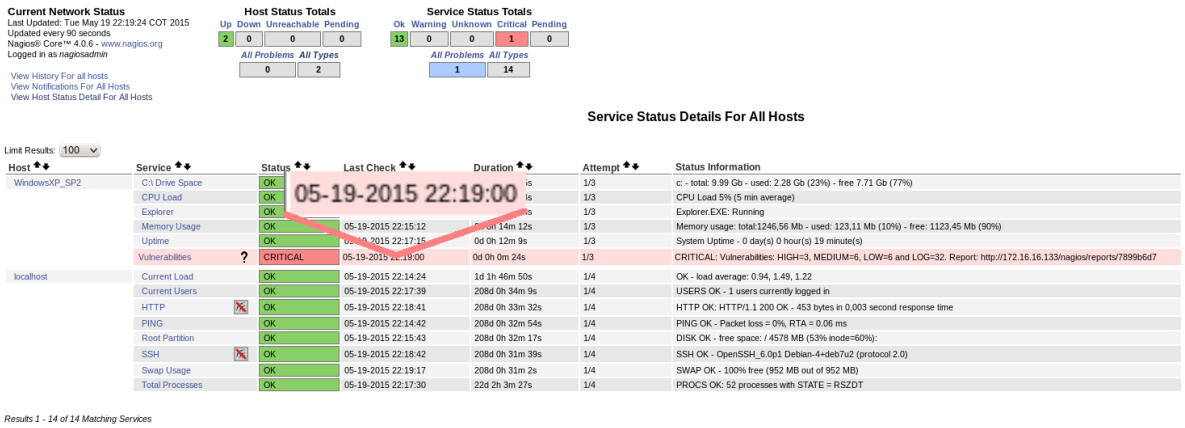
Por ello puede concluirse que la integración del sistema Nagios con el plugin check_openvas es funcional y cumple con los requerimientos planteados en el objetivo del proyecto.

**5.1.2 Tiempo de escaneo.** Para determinar el tiempo promedio de escaneo del plugin, se procede a registrar 4 tiempos consecutivos de último chequeo o *last check* mostrado en la consola web. La diferencia entre 2 registros consecutivos

dará como resultado el tiempo total que tarda en ejecutarse el plugin. Los registros de último chequeo tomados para el ejercicio se mostrarán a continuación.

La figura 21 muestra la hora de partida para la toma de tiempos.

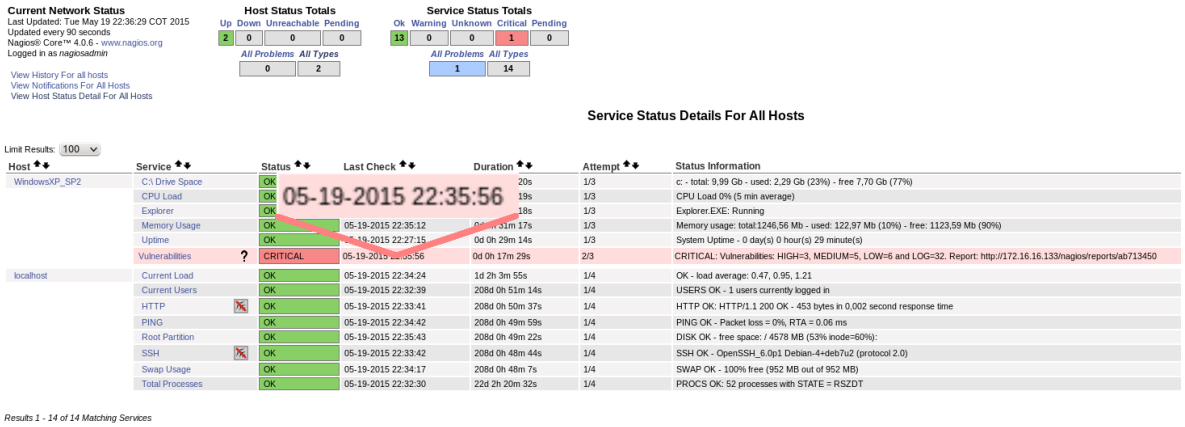
Figura 21. Registro 1 de último chequeo del plugin check_openvas.




Fuente: autor del proyecto.

La figura 22 muestra la hora en que se registra una nueva lectura de datos por parte del sistema Nagios. El tiempo registrado aquí es 22:35:56. Al restarle el tiempo de partida 22:19:00 se obtiene como resultado el tiempo de ejecución del plugin, que para este caso es 16 minutos y 56 segundos.

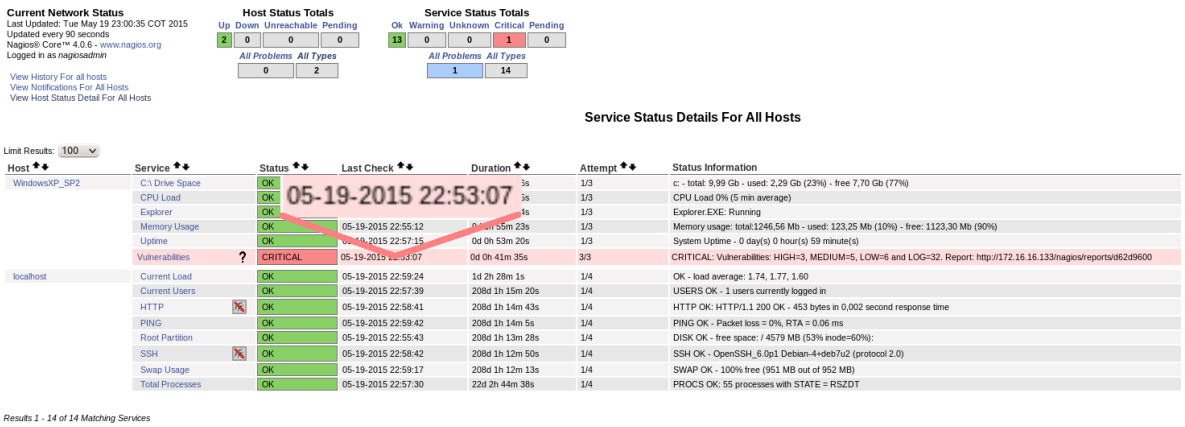
Figura 22. Registro 2 de último chequeo del plugin check_openvas.




Fuente: autor del proyecto.

En la figura 23 se evidencia el siguiente tiempo de lectura de datos, el cual es 22:53:07. Si a este tiempo se le resta el tiempo de la lectura inmediatamente anterior, o sea 22:35:56, se obtiene como resultado que el tiempo de ejecución de plugin es 17 minutos y 11 segundos.

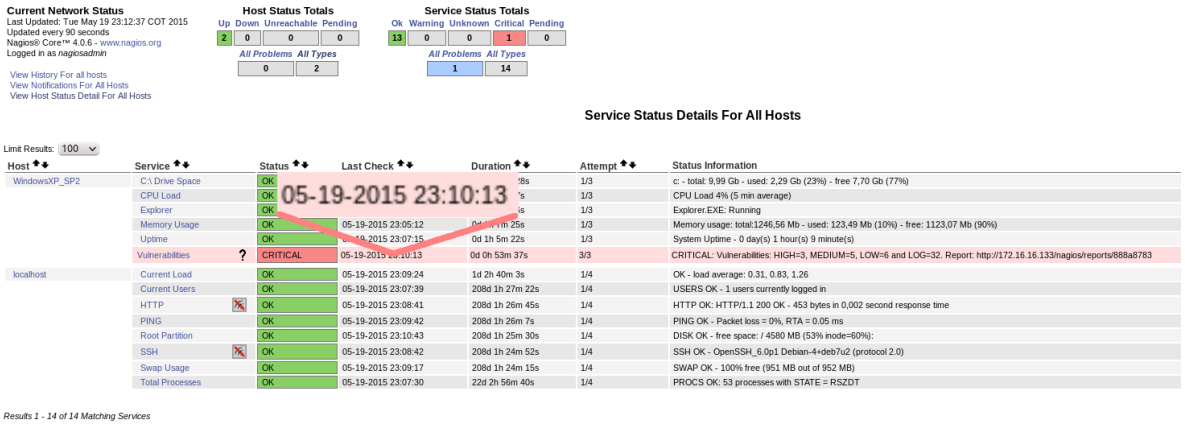
Figura 23. Registro 3 de último chequeo del plugin check_openvas.




Fuente: autor del proyecto.

En la figura 24 se evidencia el último tiempo tomado el cual se corresponde con el cuarto tiempo de lectura de datos del sistema Nagios. El tiempo registrado es 23:10:13, y al restarle el tiempo inmediatamente anterior, o sea 22:53:07, se obtiene que el tiempo de ejecución del plugin para esta ocasión es de 17 minutos y 6 segundos.

Figura 24. Registro 4 de último chequeo del plugin check_openvas.




Fuente: autor del proyecto.

Consolidando los tiempos de ejecución obtenidos, los cuales son 00:16:56, 00:17:11 y 00:17:06, podemos deducir que el plugin tarda en promedio 17 minutos y 4 segundos en ejecutarse.

**5.1.3 Vulnerabilidades detectadas.** Las vulnerabilidades detectadas para las 3 pruebas realizadas se muestran en la tabla 5.

Tabla 5. Vulnerabilidades halladas por OpenVAS.

| Nombre | Amenaza | Puerto |
|---|---|---|
| http TRACE XSS attack | High (CVSS: 5.8) | http (80/tcp) |
| Microsoft Windows SMB Server NTLM Multiple Vulnerabilities (971468) | High (CVSS: 10.0) | microsoft-ds (445/tcp) |
| Microsoft Remote Desktop Protocol Remote Code Execution Vulnerabilities (2671387) | High (CVSS: 9.3) | ms-wbt-server (3389/tcp) |
| Relative IP Identification number change | Medium (CVSS: 2.6) | general/tcp |
| Check for Apache Multiple / vulnerability | Medium (CVSS: 5.0) | http (80/tcp) |
| Apache Connection Blocking Denial of Service | Medium (CVSS: 5.0) | http (80/tcp) |
| Apache Web Server ETag Header Information Disclosure Weakness | Medium (CVSS: 4.3) | http (80/tcp) |
| Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability | Medium (CVSS: 4.3) | http (80/tcp) |
| Check for SSL Weak Ciphers | Medium (CVSS: 4.3) | netsaint (5666/tcp) |
| SMB Test | Low (CVSS: 0.0) | general/SMBClient |
| Microsoft Remote Desktop Protocol Detection | Low (CVSS: 0.0) | ms-wbt-server (3389/tcp) |
| NTP read variables | Low (CVSS: 0.0) | ntp (123/udp) |

Fuente: autor del proyecto.

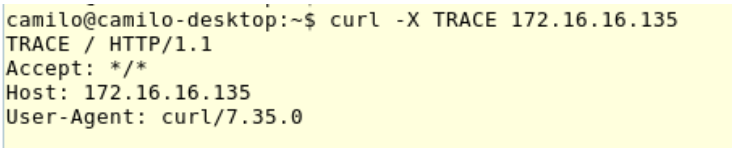
**5.1.4 Verdaderos positivos.** A continuación se enumera y detalla cada una de las vulnerabilidades que han sido reconocidas adecuadamente.

**5.1.4.1 Análisis de la vulnerabilidad “http TRACE XSS attack”.** De acuerdo a la evaluación inicial, esta vulnerabilidad es potencialmente legítima lo cual se demuestra a continuación.

De acuerdo a las investigaciones realizadas, esta vulnerabilidad es explotada por el ataque conocido como *Cross Site Tracing (XST)*. Dicho ataque implica el uso de *Cross-site Scripting (XSS)* y los métodos *HTTP TRACE* o *TRACK HTTP*. El método *TRACE* permite al cliente ver lo que está siendo recibido en el otro extremo de la cadena de petición y usar esos datos para realizar pruebas o información de diagnóstico; el método TRACK funciona de la misma manera, pero es específico del servidor web Microsoft IIS. XST podría ser utilizado como un método para robar las cookies del usuario a través de Cross-site Scripting (XSS)[10].

De acuerdo con el sitio web de *Open Web Application Security Project OWASP,* el comando que se muestra en la figura 25 es usado para enviar una petición *TRACE* a un servidor web. Para este caso, la petición se ha enviado al host objetivo para obtener su respuesta.

Figura 25. Envío de petición TRACE al servidor web del host objetivo.




Fuente: autor del proyecto.

Como puede observarse, el servidor web del host objetivo tiene activo estos métodos, por lo cual es explotable por el ataque en mención y confirma a su vez que la vulnerabilidad se trata de un verdadero positivo.

**5.1.4.2 Boletín de Seguridad Microsoft MS10-012.** Este boletín se corresponde con la vulnerabilidad nombrada como *Microsoft Windows SMB Server NTLM Multiple Vulnerabilities (971468)*. De acuerdo a este boletín[11], aquí se reúnen 4 vulnerabilidades las cuales afectan, con distinta criticidad, al protocolo del bloque

10 [Citado en 24 de Junio de 2015] Disponible en < https://www.owasp.org/index.php/Cross_Site_Tracing >.

11 [Citado en 28 de Junio de 2015] Disponible en < https://technet.microsoft.com/es-es/library/security/ms10-012.aspx >.

de mensajes del servidor (SMB) de Microsoft. Las vulnerabilidades que componen dicho boletín son:

- Vulnerabilidad de desbordamiento del nombre de ruta en SMB (CVE-2010-0020).
- Vulnerabilidad de daños en la memoria relacionada con SMB (CVE-2010-0021).
- Vulnerabilidad de puntero nulo en SMB (CVE-2010-0022).
- Vulnerabilidad de falta de entropía en la autenticación NTLM en SMB (CVE-2010-0231).

Un atacante que pueda aprovechar alguna de estas vulnerabilidades puede realizar desde una denegación del servicio hasta la ejecución de código remoto, dependiendo de la vulnerabilidad que sea explotada.

De acuerdo a la información del boletín, existen varios métodos de detección los cuales consisten en el uso de herramientas especializadas como *Microsoft Baseline Security Analyzer* o *Windows Server Update Services* entre otras. Sin embargo, estas herramientas requieren ser instaladas en el host objetivo, lo cual puede alterar las vulnerabilidades ya detectadas o incluso, puede introducir nuevas vulnerabilidades al sistema. Otro método que no está directamente informado en el boletín pero que es igualmente válido, consiste en evaluar las versiones de los archivos instalados por la actualización. Este método es elegido por ser el que menor invasión presenta al host objetivo.

La actualización de seguridad que corrige dichas vulnerabilidades en Windows XP es la KB971468,la cual actualiza el archivo Srv.sys a la versión 5.1.2600.3662 con fecha del 31 de diciembre de 2009 conforme se comenta en la tabla 6.

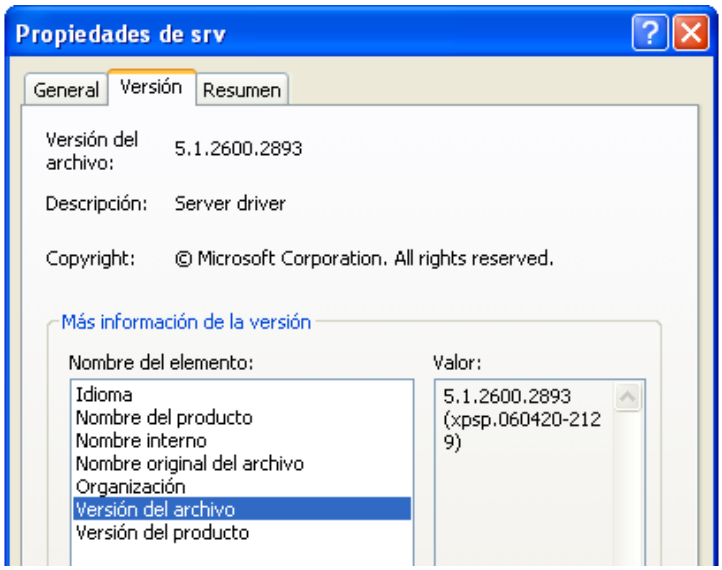
Tabla 6. Versiones de archivos de actualización KB971468 para Windows XP.

| File name | File version | File size | Date | Time | Platform | SP requirement | Service branch |
|---|---|---|---|---|---|---|---|
| Srv.sys | 5.1.2600.3662 | 352,640 | 31-Dec-2009 | 16:14 | x86 | SP2 | SP2GDR |
| Srv.sys | 5.1.2600.3662 | 352,640 | 31-Dec-2009 | 15:06 | x86 | SP2 | SP2QFE |
| Srv.sys | 5.1.2600.5923 | 353,792 | 31-Dec-2009 | 16:50 | x86 | SP3 | SP3GDR |
| Srv.sys | 5.1.2600.5923 | 353,792 | 01-Jan-2010 | 07:58 | x86 | SP3 | SP3QFE |

Fuente: https://support.microsoft.com/en-us/kb/971468.

Al verificar la versión del archivo que está instalado en el host objetivo, se encuentra que el sistema cuenta con una versión anterior a la que instala la actualización KB971468. En la figura 26 se muestran los detalles de dicho archivo.

Figura 26. Versión del archivo Srv.sys del host objetivo.




Fuente: autor del proyecto.

El archivo Srv.sys encontrado en el host objetivo concuerda con el desplegado por la actualización KB917159 creada para solucionar las vulnerabilidades informadas en el boletín de seguridad Microsoft MS06-035, el cual data del 11 de Julio de 2006 como se evidencia en su sitio web[12], tiempo anterior al que se conocieron las vulnerabilidades contenidas en el boletín de seguridad Microsoft MS10-0012 que data del año 2010.

Conforme la evidencias adquiridas, se concluye que la vulnerabilidad en análisis es legítima y sí afecta al host objetivo.

**5.1.4.3 Boletín de Seguridad Microsoft MS12-020.** El boletín en mención hace referencia a la vulnerabilidad nombrada como *Microsoft RDP Remote Code Execution Vulnerabilities (2671387)*.

12 [Citado en 28 de Junio de 2015] Disponible en < https://www.microsoft.com/spain/technet/seguridad/boletines/ms06-035-IT.mspx >.

Este boletín hace referencia a 2 vulnerabilidades independientes pero relacionadas, las cuales son:

- Vulnerabilidad de protocolo de escritorio remoto (CVE-2012-0002).
- Vulnerabilidad de denegación de servicio en Terminal Server (CVE-2012-0152).

Un atacante que pueda aprovechar una de estas dos vulnerabilidades, podría realizar desde una denegación del servicio hasta la ejecución de código remoto, dependiendo de vulnerabilidad que sea explotada.

Al igual que con la vulnerabilidad analizada anteriormente, recurrimos a evaluar nuevamente las versiones de los archivos que contiene la actualización que corrige estas vulnerabilidades, y a compararlo posteriormente con la versión instalada el host objetivo, con el fin de determinar si el sistema es vulnerable o no.

La actualización de seguridad que corrige dicha vulnerabilidad es KB2621440, la cual actualiza el archivo Rdpwd.sys de Windows XP a la versión 5.1.2006.6187, tal como se indica en la tabla 7.

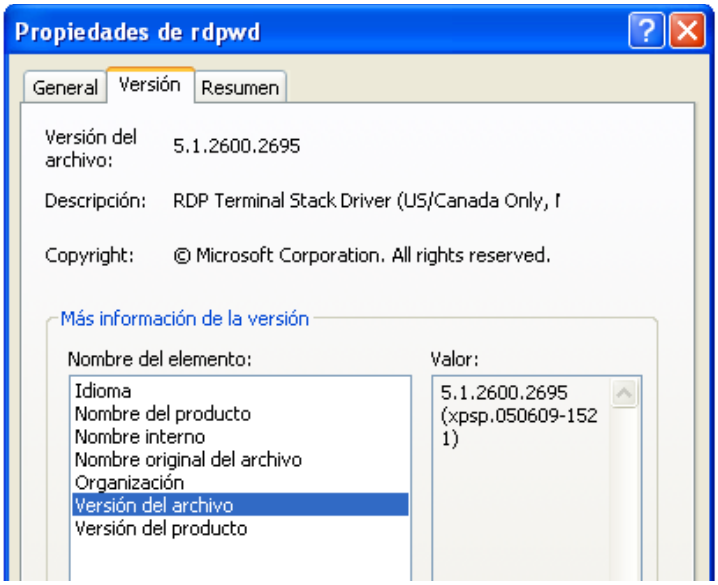
Tabla 7. Versiones de archivos de actualización KB2621440 para Windows XP.

| File name | File version | File size | Date | Time | Hashes | Platform | SP requirement | Service branch |
|---|---|---|---|---|---|---|---|---|
| Rdpwd.sys | 5.1.2600.6187 | 139,784 | 09-Jan-2012 | 16:20 | **MD5:** 5B3055DAA788BD688594D2F5981F2A83 **SHA1:** 139454B10FBFC2717E54696A7ABC64B571BA6E8B | x86 | SP3 | SP3GDR |
| Rdpwd.sys | 5.1.2600.6187 | 139,784 | 09-Jan-2012 | 16:19 | **MD5:** 2D293B720C206473A05950CE007DB12A **SHA1:** 9AD71C2949AB1F928D88D34717ED39AE4675A72E | x86 | SP3 | SP3QFE |

Fuente: https://support.microsoft.com/en-us/kb/2621440.

Si se observa la versión del archivo en el host objetivo, se evidencia que éste se encuentra en una versión anterior. En la figura 27 se muestran los detalles de dicho archivo.

Figura 27. Versión del archivo Rdpwd.sys del host objetivo.




Fuente: autor del proyecto.

La versión evidenciada en el host objetivo corresponde a la actualización KB899591 creada pasa solucionar las vulnerabilidades informadas en el boletín de seguridad Microsoft MS05-041, el cual data del 09 de Agosto de 2005 como se informa en el sitio web de Microsoft[13], tiempo anterior al que se dio a conocer las vulnerabilidades contenidas en el boletín de seguridad Microsoft MS12-020 que data del año 2012.

Conforme las evidencias adquiridas, se concluye que la vulnerabilidad en mención es legítima y sí afecta al host objetivo.

**5.1.4.4 Análisis de "Relative IP Identification number change".** De acuerdo al reporte obtenido de OpenVAS, esta vulnerabilidad hace referencia a que el host objetivo no usa identificadores IP aleatorios, por lo que es posible predecir el siguiente valor del campo ip_id de los paquetes IP enviados por el host. Esta vulnerabilidad puede ser usada por un atacante para realizar un *blind portscan*, para determinar la cantidad de peticiones que atiende un servidor en ciertos tiempos del día y así enfocar sus esfuerzos en máquinas críticas, o incluso

13 [Citado en 28 de Junio de 2015] Disponible en < https://technet.microsoft.com/en-us/library/security/ms05-041.aspx >.

determinar el número de solicitudes que atiende un servidor web en cierto instante de tiempo.

Para poder determinar si esta vulnerabilidad es legítima, es necesario usar un sniffer de red para poder identificar el campo ip_id y determinar si efectivamente este identificador es predecible.

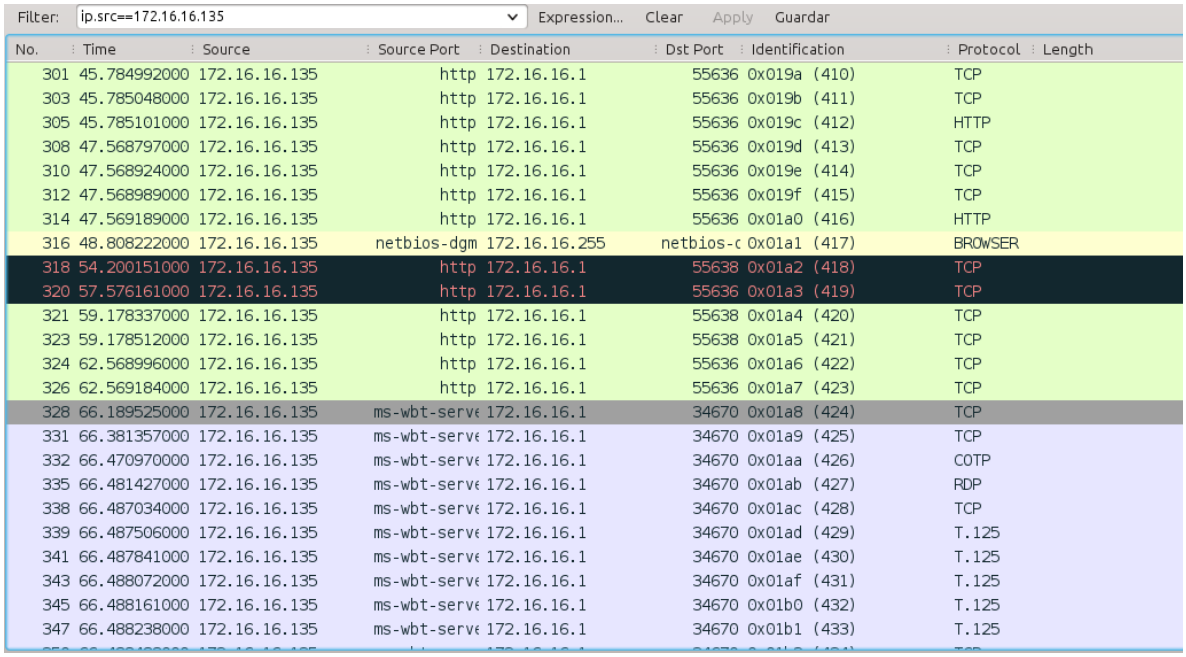
La prueba se realizará de la siguiente manera:

- Es necesario disponer de dos máquinas. La máquina atacante y el host objetivo.
- En la máquina atacante se ejecuta un sniffer para capturar el tráfico. Para este caso se usa el sniffer de red conocido con el nombre de Wireshark
- Posteriormente se procede a ingresar vía web al servidor Apache y realizar solicitudes de algunas URL.
- Luego se procede a realizar una petición mediante el protocolo RDP al puerto 3389.
- Por último se procede a analizar la traza del sniffer obtenida.

Los resultados se pueden observar en la figura 28, la cual muestra un extracto de la herramienta Wireshark, filtrada por la IP de origen. Allí se evidencia claramente que los paquetes enviados por el host objetivo son consecutivos (comenzando por el ip_id 410 y finalizando en el ip_id 433) y por lo tanto se puede concluir que el campo ip_id es predecible.

Con la prueba realizada se concluye que la vulnerabilidad analizada corresponde a otra vulnerabilidad legítima del host objetivo.

Figura 28. Identificador IP de una trama de paquetes del host objetivo.



Filter: ip.src==172.16.16.135 Expression... Clear Apply Guardar

| No. | Time | Source | Source Port | Destination | Dst Port | Identification | Protocol | Length |
|---|---|---|---|---|---|---|---|---|
| 301 | 45.784992000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019a (410) | TCP | |
| 303 | 45.785048000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019b (411) | TCP | |
| 305 | 45.785101000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019c (412) | HTTP | |
| 308 | 47.568797000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019d (413) | TCP | |
| 310 | 47.568924000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019e (414) | TCP | |
| 312 | 47.568989000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x019f (415) | TCP | |
| 314 | 47.569189000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x01a0 (416) | HTTP | |
| 316 | 48.808222000 | 172.16.16.135 | netbios-dgm | 172.16.16.255 | netbios-c | 0x01a1 (417) | BROWSER | |
| 318 | 54.200151000 | 172.16.16.135 | http | 172.16.16.1 | 55638 | 0x01a2 (418) | TCP | |
| 320 | 57.576161000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x01a3 (419) | TCP | |
| 321 | 59.178337000 | 172.16.16.135 | http | 172.16.16.1 | 55638 | 0x01a4 (420) | TCP | |
| 323 | 59.178512000 | 172.16.16.135 | http | 172.16.16.1 | 55638 | 0x01a5 (421) | TCP | |
| 324 | 62.568996000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x01a6 (422) | TCP | |
| 326 | 62.569184000 | 172.16.16.135 | http | 172.16.16.1 | 55636 | 0x01a7 (423) | TCP | |
| 328 | 66.189525000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01a8 (424) | TCP | |
| 331 | 66.381357000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01a9 (425) | TCP | |
| 332 | 66.470970000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01aa (426) | COTP | |
| 335 | 66.481427000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01ab (427) | RDP | |
| 338 | 66.487034000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01ac (428) | TCP | |
| 339 | 66.487506000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01ad (429) | T.125 | |
| 341 | 66.487841000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01ae (430) | T.125 | |
| 343 | 66.488072000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01af (431) | T.125 | |
| 345 | 66.488161000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01b0 (432) | T.125 | |
| 347 | 66.488238000 | 172.16.16.135 | ms-wbt-serve | 172.16.16.1 | 34670 | 0x01b1 (433) | T.125 | |

Fuente: autor del proyecto.

**5.1.4.5 Análisis de "Check for Apache Multiple / vulnerability".** Esta vulnerabilidad hace referencia al CVE-2000-0505[14], el cual informa que permite a un atacante remoto listar el contenido del directorio en el servidor HTTP Apache 1.3.x para plataformas Windows, realizando una solicitud de URL conteniendo un alto número de slashes. La referencia también informa que la solución a dicha vulnerabilidad está integrada en el servidor HTTP Apache versión 1.3.14, lo cual hace pensar que éste puede tratarse del primer falso positivo reconocido por OpenVAS.

Adicional a la información de la referencia, en el propio reporte de OpenVAS se comenta que 197 slashes provocan esta vulnerabilidad.

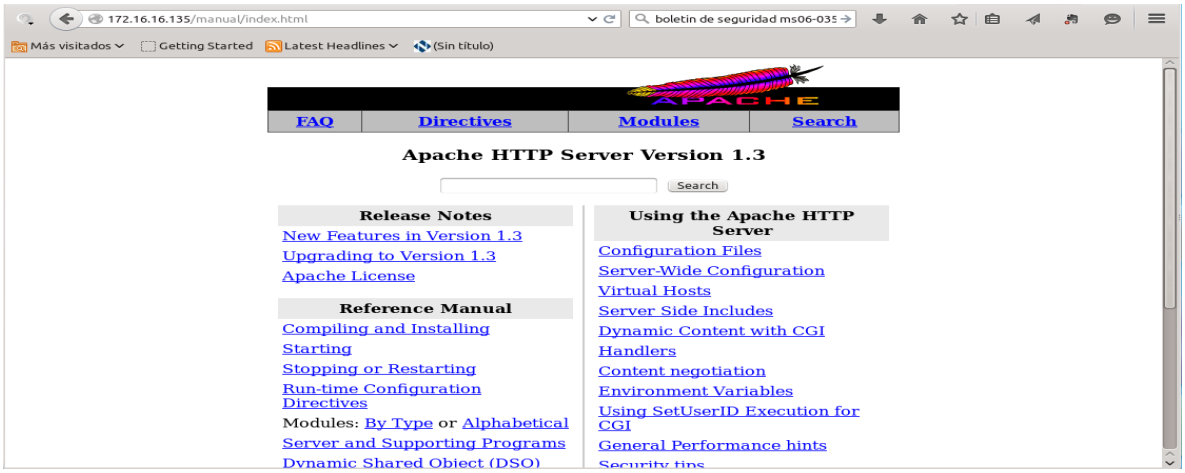
Debido a la característica de la vulnerabilidad y su fácil aprovechamiento, se procede a realizar pruebas con la información obtenida para explotarla.

Para evaluar esta vulnerabilidad, se ingresa al servidor web del host objetivo apuntando al sitio que contiene la documentación de apache propia del servidor,

14 [Citado en 28 de Junio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2000-0505 >.

cuya URL es http://172.16.16.135/manual/. La página que carga se muestra en la figura 29.

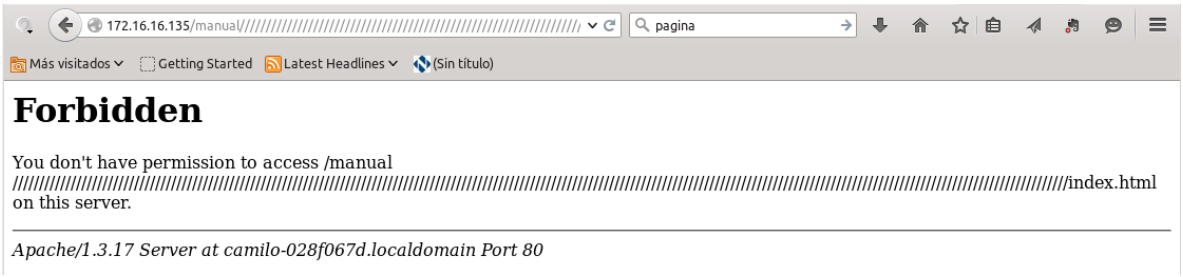
Figura 29. Página de documentación de Apache del host objetivo.




Fuente: autor del proyecto.

Se procede entonces a ingresar 197 slashes al final de la URL. El resultado al intentar acceder a esta nueva URL se muestra en la figura 30.

Figura 30. Resultado de intento de acceso a URL con 197 slashes.




Fuente: autor del proyecto.

Como se evidencia, a pesar que la salida del servidor HTTP Apache es diferente, no se evidencia la vulnerabilidad. Sin embargo, este cambio de salida genera la duda de la cantidad de slashes requeridos para producir el cambio de salida; por ello se procede a realizar pruebas de ensayo y error con cantidades comprendidas entre 2 y 196 slashes. El resultado se resume en la tabla 8.

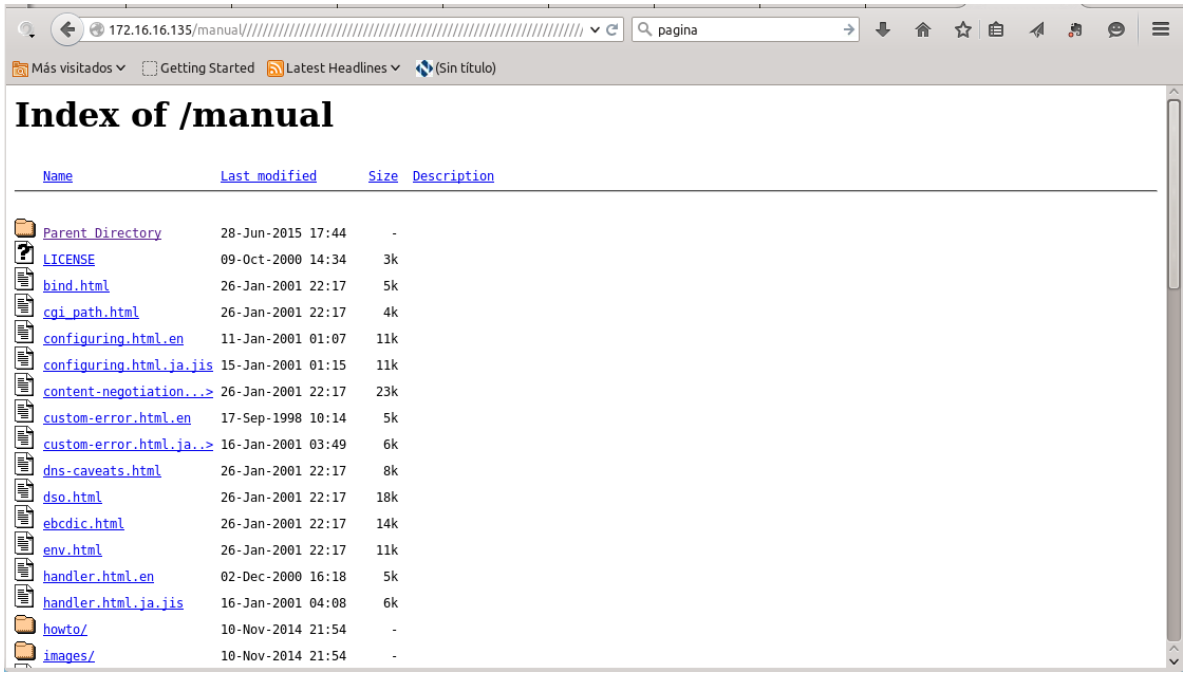
Tabla 8. Pruebas de ensayo y error de vulnerabilidad CVE-2000-0505.

| Slashes | Salida |
|---|---|
| 1 a 189 | Index.html |
| 190 a 192 | Contenido de directorio |
| 193 a 197 | Mensaje de prohibido |

Fuente: autor del proyecto.

De acuerdo a la tabla 8, sí es posible obtener el contenido del directorio pero no con 197 slashes tal como lo afirma el reporte de OpenVAS, sino con valores comprendidos entre 190 y 192 slashes. La figura 31 muestra la salida de la URL finalizando con 190, 191 y 192 slashes. Tal contenido es el mismo que se encuentra alojado en la ruta "c:\Archivos de programa\Apache Group\Apache\htdocs\manual\" del host objetivo.

Figura 31. Resultado de intento de acceso a URL con 190 a 192 slashes.




Fuente: autor del proyecto.

Las pruebas realizadas junto con los resultados obtenidos confirman que la vulnerabilidad en mención sí existe en el host objetivo.

**5.1.4.6 Análisis de “Check for SSL Weak Ciphers”.** La vulnerabilidad nombrada como *Check for SSL Weak Ciphers* hace referencia a la detección de cifrados SSL débiles en el puerto TCP 5666, sobre el cual escucha el agente NRPE *Nagios Remote Plugin Executor.*

Un cifrado débil es aquel que usa claves menores a 128 bits[15]. Además, pueden incluirse en este grupo los algoritmos como MD5, los cuales a la fecha son considerados como débiles. Para detectar la presencia de dichos cifrados, se usará la herramienta de código abierto SSLScan, la cual se encarga de consultar los cifrados soportados por un servicio SSL.

La herramienta se instala en un equipo atacante y es lanzada de tal manera que consulte el puerto TCP 5666 del host objetivo. El comando usado es “sslscan --no-failed 172.16.16.135:5666” y el resultado puede visualizarse en la figura 32.

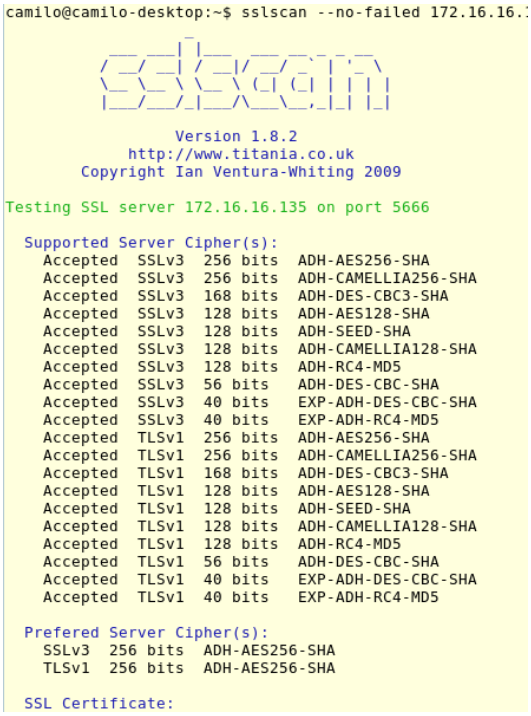
Tal como puede verse, la herramienta SSLScan reconoce los siguientes cifrados que pueden clasificarse como débiles:

- SSLv3 128 bits ADH-RC4-MD5.
- SSLv3 56 bits ADH-DES-CBC-SHA.
- SSLv3 40 bits EXP-ADH-DES-CBC-SHA.
- SSLv3 40 bits EXP-ADH-RC4-MD5.
- TLSv1 128 bits ADH-RC4-MD5.
- TLSv1 56 bits ADH-DES-CBC-SHA.
- TLSv1 40 bits EXP-ADH-DES-CBC-SHA.
- TLSv1 40 bits EXP-ADH-RC4-MD5.

Con los resultados obtenidos de la herramienta SSLScan, es posible identificar esta vulnerabilidad como legítima.

15 [Citado en 02 de Agosto de 2015] Disponible en < https://www.owasp.org/index.php/Testing_for_Weak_SSL/TLS_Ciphers,_Insufficient_Transport_Layer_Protection_%28OTG-CRYPST-001%29#Weak_SSL.2FTLS_Ciphers.2FProtocols.2FKeys >.

Figura 32. Cifrados soportados por NRPE.




Fuente: autor del proyecto.

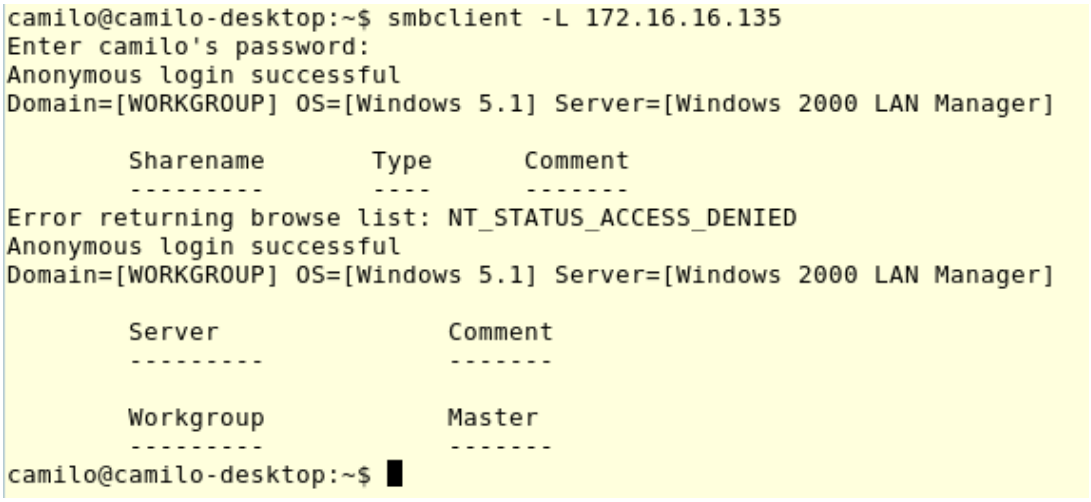
**5.1.4.7 Análisis de “SMB Test”.** El *Server Messages Block* es un protocolo de red usado para compartir archivos en una red. Este protocolo es implementado en los sistemas operativos Microsoft Windows y es conocido como *Microsoft SMB Protocol*[16].

Al ser identificado el servicio que se ejecuta en el host objetivo, representa una vulnerabilidad que a pesar de clasificarse como baja, puede representar una grieta donde puede vulnerarse el sistema.

Para identificar si el servicio verdaderamente se ejecuta en el host objetivo, se usará el comando smbclient el cual es una herramienta que hace parte de la suite samba. Éste es un cliente que puede “hablar” con un servidor SMB. Para la prueba, es necesario ejecutar el comando “smbclient -L 172.16.16.135” desde un equipo atacante que tenga la suite samba previamente instalada. El resultado de la prueba puede verse en la figura 33.

16 [Citado en 02 de Agosto de 2015] Disponible en < https://msdn.microsoft.com/en-us/library/windows/desktop/aa365233%28v=vs.85%29.aspx >.

Figura 33. Conexión de un atacante al protocolo SMB.




Fuente: autor del proyecto.

Como puede verse, no es necesario conocer contraseña alguna, ya que al realizar un acceso como usuario Anonymous, se obtienen los siguientes datos:

- Domain=[WORKGROUP]
- OS=[Windows 5.1]
- Server=[Windows 2000 LAN Manager]

Con los datos obtenidos rápidamente puede identificarse el sistema operativo Windows 5.1, el cual se corresponde con Windows XP, conforme al listado de versiones del sistema operativo Windows ofrecido en internet por Microsoft[17].

**5.1.4.8 Análisis de "Microsoft Remote Desktop Protocol Detection".** El *Microsoft Remote Desktop Protocol*, conocido también como RDP, provee visualización remota y la capacidad de acceso a las aplicaciones que corren en un servidor por medio de una conexión de red[18]. Por su naturaleza, se considera una

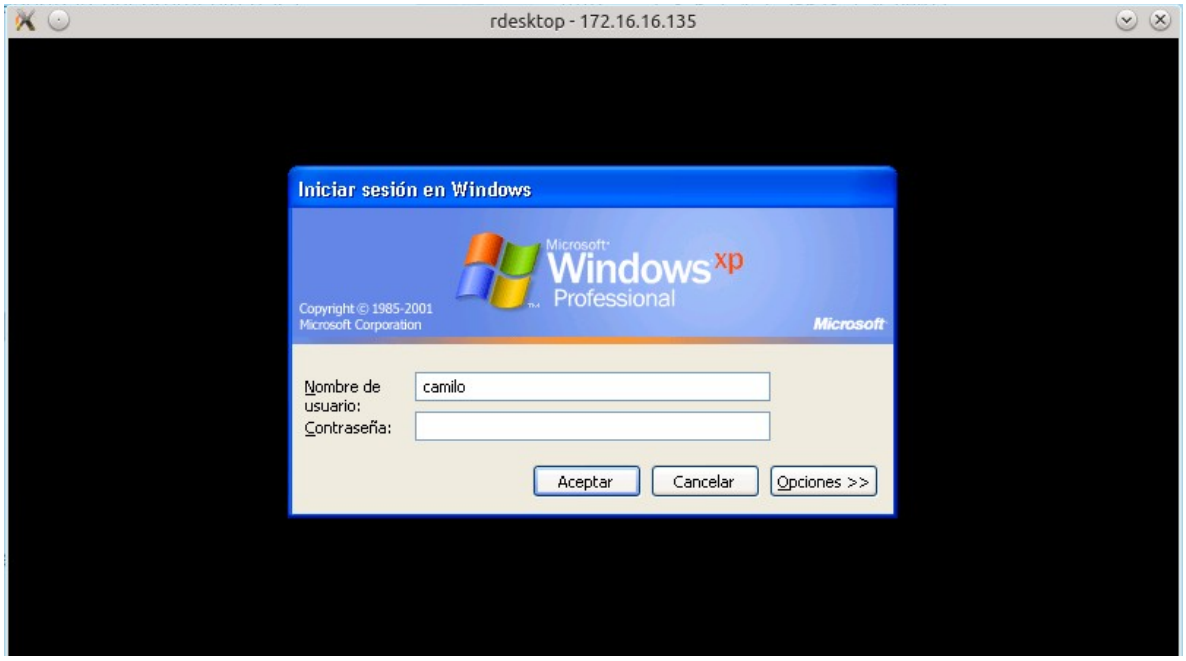
17 [Citado en 02 de Agosto de 2015] Disponible en < https://msdn.microsoft.com/en-us/library/windows/desktop/ms724832%28v=vs.85%29.aspx >.

18 [Citado en 02 de Agosto de 2015] Disponible en < https://msdn.microsoft.com/es-es/library/aa383015%28v=vs.85%29.aspx >.

vulnerabilidad que el protocolo sea alcanzado por un atacante que pueda romper su seguridad para ganar acceso al host objetivo.

Un atacante puede acceder a este protocolo usando cualquier cliente que lo soporte. Para este caso, se usa el cliente Linux rdesktop, el cual se utiliza para realizar conexiones RDP a equipos con el sistema operativo Windows con el servicio activo. Un intento de conexión con el cliente rdesktop desde un atacante hacia el host objetivo, arroja la ventana de acceso al sistema que se muestra en la figura 34.

Figura 34. Conexión RDP desde el atacante.




Fuente: autor del proyecto.

Con la anterior prueba se verifica que el protocolo si está habilitado en el host objetivo.

**5.1.4.9 Análisis de "NTP read variables".** Al igual que la vulnerabilidad anterior, ésta solo se refiere a que se ha detectado un servicio NTP ejecutándose en el puerto UDP 123. Aunque el servicio NTP se usa para temas relacionados con la sincronía del tiempo, en algunos casos puede proporcionar información adicional la cual se considera innecesaria e insegura.

Para validar que efectivamente está corriendo un servidor NTP en el host objetivo, se realizan las siguientes pruebas desde el equipo atacante:

• Se comprueba que el puerto UDP 123 este escuchado en el host objetivo. Para ello se usa el comando netcat tal como se muestra en la figura 35.

Figura 35. Comprobación de escucha del puerto UDP 123.

```
camilo@camilo-desktop:~$ netcat -nuv 172.16.16.135 123
Connection to 172.16.16.135 123 port [udp/*] succeeded!
```

Fuente: autor del proyecto.

• Se consulta la información proporcionada por el servicio. Para ello se usan dos herramientas: nmap y ntpq. La consulta con nmap puede verse en la figura 36, mientras que la consulta con ntpq puede verse en la figura 37.

Figura 36. Consulta al servicio NTP con nmap.

```
camilo@camilo-desktop:~$ sudo nmap -sU -p 123 --script ntp-info 172.16.16.135

Starting Nmap 6.40 ( http://nmap.org ) at 2015-08-03 19:56 COT
Nmap scan report for 172.16.16.135
Host is up (0.00019s latency).
PORT    STATE SERVICE
123/udp open  ntp
| ntp-info:
|_  receive time stamp: 2015-08-04T00:56:44
MAC Address: 00:0C:29:90:B6:72 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 5.77 seconds
```

Fuente: autor del proyecto.

Figura 37. Consulta al servicio NTP con ntpq.

```
camilo@camilo-desktop:~$ ntpq -c rv 172.16.16.135
172.16.16.135: timed out, nothing received
***Request timed out
```

Fuente: autor del proyecto.

Como se evidencia en los resultados de las pruebas, el puerto UDP 123 está en escucha, y a pesar que no se obtiene información innecesaria, se puede observar que sobre este puerto se ejecuta un servicio NTP tal como lo muestra la prueba con nmap.

**5.1.5 Falsos positivos.** A continuación se mencionan las vulnerabilidades las cuales no pueden ser explotadas, pero que el sistema ha reconocido como explotables.

**5.1.5.1 Análisis de "Apache Connection Blocking Denial of Service".** Esta vulnerabilidad ha tenido una dificultad especial para su análisis debido a la ambigua información que ese encuentra sobre ésta.

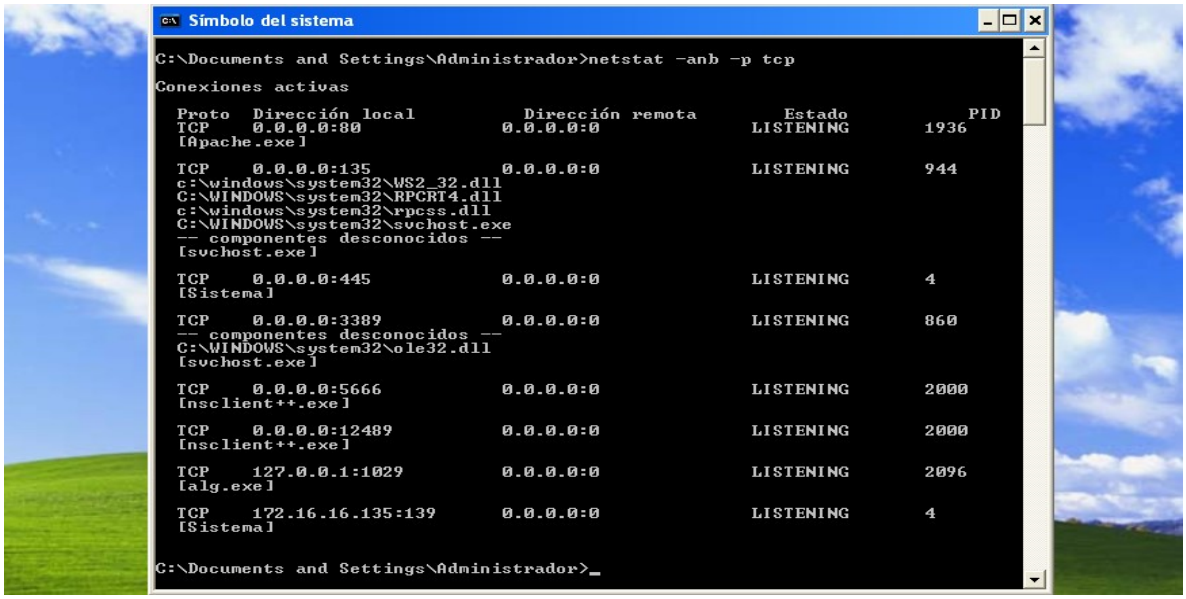
Dicha vulnerabilidad permite a un atacante remoto provocar una denegación de servicio (bloqueo de nuevas conexiones) a través de una conexión de corta duración en un socket raramente accedido del servidor HTTP Apache en versiones anteriores a la 1.3.31 y 2.0.49[19]. Lo particular de esta vulnerabilidad es que se presenta cuando se está usando múltiples sockets de escucha en algunas plataformas (de los cuales no fue posible descartar el sistema operativo del host objetivo con la información consultada).

Como no fue posible explotar dicha vulnerabilidad, la clave para determinar o no la veracidad de la vulnerabilidad está en la propia descripción de la misma, justo en donde menciona que se presenta cuando se está usando múltiples sockets. Este enunciado permite clasificar la vulnerabilidad como falso positivo, ya que el servidor HTTP Apache se ha instalado con las opciones por defecto y solamente está escuchando por un único socket, conforme se evidencia en la figura 38.

A pesar que la versión del servidor HTTP Apache es afectada por la vulnerabilidad analizada, esta no se activará a menos que se realicen cambios en su configuración para la habilitación de múltiples sockets. Por ello, podría considerarse que la vulnerabilidad esta mitigada y no es explotable.

19 [Citado en 05 de Julio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-20 >

Figura 38. Sockets de escucha del host objetivo.




Fuente: autor del proyecto.

**5.1.5.2 Análisis de “Apache WS ETag Header Info Disclosure Weakness”.** Esta vulnerabilidad hace referencia a que un atacante remoto puede obtener información sensible a través de la cabecera ETag. Esta cabecera puede revelar el número de inodo, o el limite MIME multiparte, que revelan los identificadores de proceso (PID)[20].

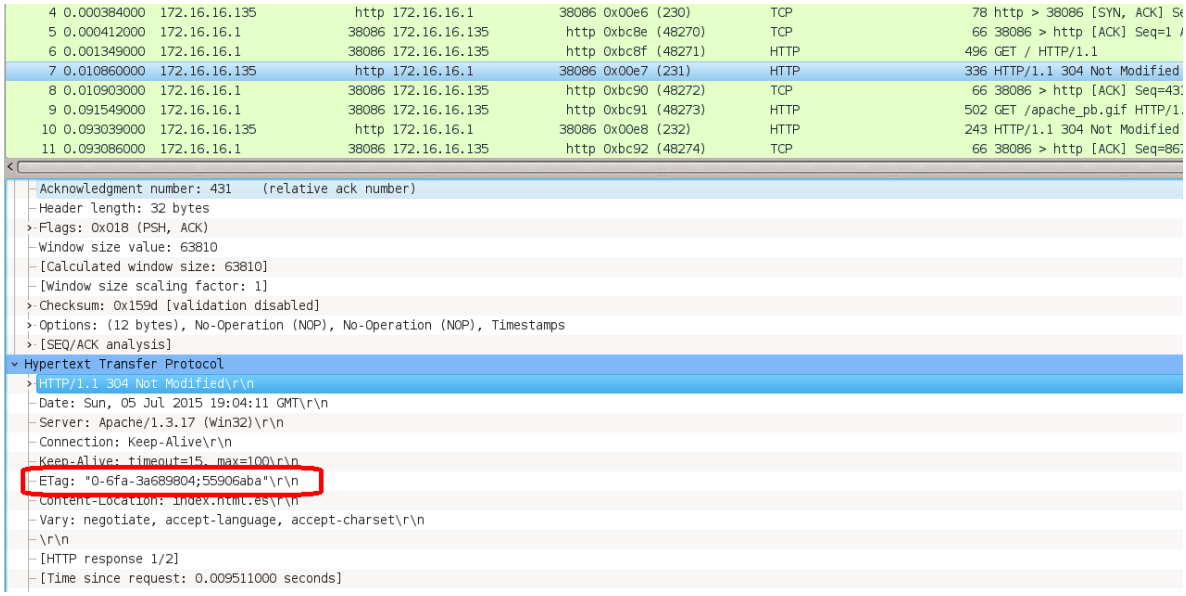
Para visualizar lo revelado por la cabecera Etag que emite el servidor HTTP Apache del host objetivo, se procede nuevamente a hacer uso del sniffer Wireshark. El procedimiento realizado es el siguiente:

- Es necesario disponer de dos máquinas. La máquina atacante y el host objetivo.
- En la máquina atacante se ejecuta el ya conocido sniffer Wireshark.
- Desde la máquina atacante se procede a realizar la solicitud de la página http://172.16.16.135, la cual corresponde al servidor HTTP Apache del host objetivo.
- Luego de cargada la página, se procede a actualizarla presionando el botón F5.

20 [Citado en 05 de Julio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2003-1418 >.

Realizados estos pasos, se obtiene la captura en el sniffer, de la cual puede observarse un extracto en la figura 39. En ésta claramente se observa que el valor de la cabecera ETag es "0-6fa-3a689804;55906aba".

Figura 39. Cabecera ETag de respuesta del host objetivo.




Fuente: autor del proyecto.

De acuerdo a la documentación del servidor HTTP Apache versión 1.3.X, esta cabecera siempre está compuesta para versiones 1.3.22 o anteriores por el inodo de archivo, el tamaño o peso del archivo, y la fecha de última modificación (mtime)[21]; dicha estructura corresponde con la versión 1.3.17 que está siendo ejecutada en el host objetivo.

De acuerdo a la información hallada y al análisis realizado sobre el tráfico capturado en el sniffer, se encuentra que la vulnerabilidad evaluada corresponde a un falso positivo porque:

- De acuerdo al reporte proporcionado por OpenVAS, el inodo obtenido es cero, el cual corresponde a un número de inodo reservado y no a un archivo como tal. El reporte completo puede consultarse en anexo A.

21 [Citado en 05 de Julio de 2015] Disponible en < http://httpd.apache.org/docs/1.3/mod/core.html#fileetag >.

• El host objetivo, por tratarse de una maquina ejecutando el sistema operativo Windows XP, usa el sistema de archivos NTFS el cual no se basa en inodos.

Por ello, la información proporcionada por la cabecera ETag no es de mucha utilidad para un atacante ya que no revela el inodo del archivo que se solicita.

**5.1.5.3 Análisis de "Apache HTTP Server 'httpOnly' Cookie Info Disclosure".** Esta vulnerabilidad hace referencia al CVE-2012-0053[22], el cual informa que el servidor HTTP Apache en las versiones comprendidas entre 2.2.0 y 2.2.21 no restringe apropiadamente la información durante la construcción del error de respuesta código de estado 400, que podría permitir a atacantes remotos obtener valores de cookie con httponly activo.

Para evaluar esta vulnerabilidad fue necesario realizar una serie de modificaciones al host objetivo. Con el fin de no afectar ni alterar los resultados obtenidos hasta el momento, se han tomado las medidas necesarias para poder restaurar el host objetivo a su estado inicial, esto aprovechando la característica de toma de snapshots que incluye la herramienta VMWare Workstation, lo cual permite restaurar el estado de una máquina virtual a un estado anterior previamente guardado.

Basado en la información publicada por el Sr. Jonathan Simon Prates disponible en internet[23], se procede a reproducir la explotación de la vulnerabilidad siguiendo la planeación siguiente:

• Instalación y configuración de una versión del servidor HTTP Apache afectada en el host objetivo. Para este caso se instala la versión 2.2.21.

• Instalación y configuración de perl en el host objetivo con el fin de poder ejecutar código perl en el servidor HTTP Apache. Para este caso se instala el software ActivePerl versión 5.20.2.2001.

• Configuración del servidor HTTP Apache para las pruebas de explotación.

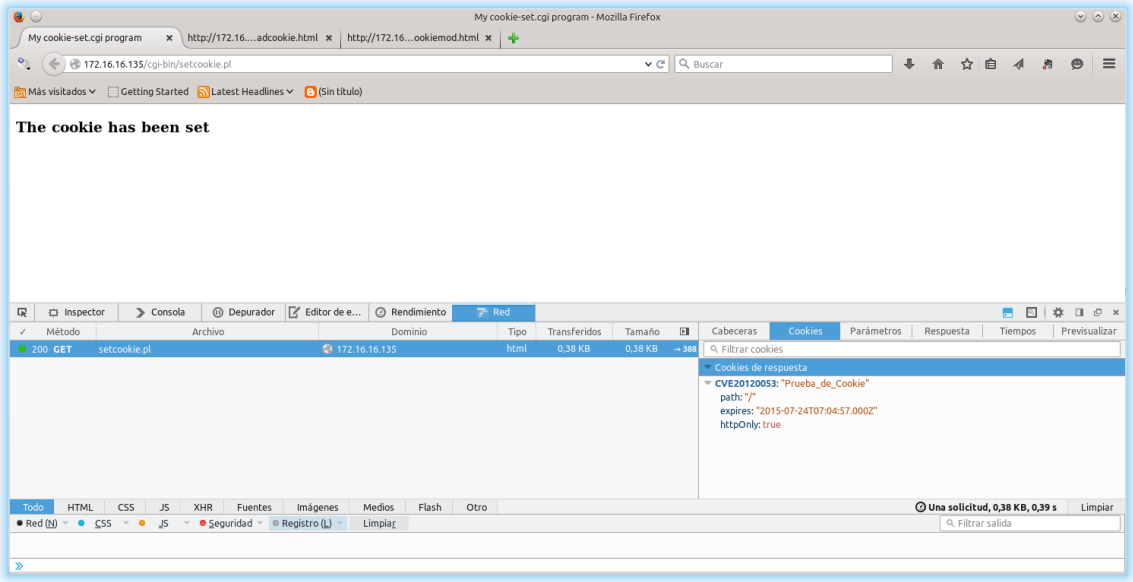
22 [Citado en 23 de Julio de 2015] Disponible en < https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-0053 >.

23 [Citado en 23 de Julio de 2015] Disponible en < https://github.com/jonathansp/CVE20120053Demo >.

Una vez realizado lo anterior, se continúa con el siguiente procedimiento utilizando un navegador web apuntando al host objetivo con el servidor HTTP Apache versión 2.2.21 activo.

- Ejecutar el script http://172.16.16.135/cgi-bin/setcookie.pl. Al ejecutar este script, se creará una cookie y se almacenará localmente tal como se muestra en la figura 40. El código del script puede ser consultado en el anexo C.

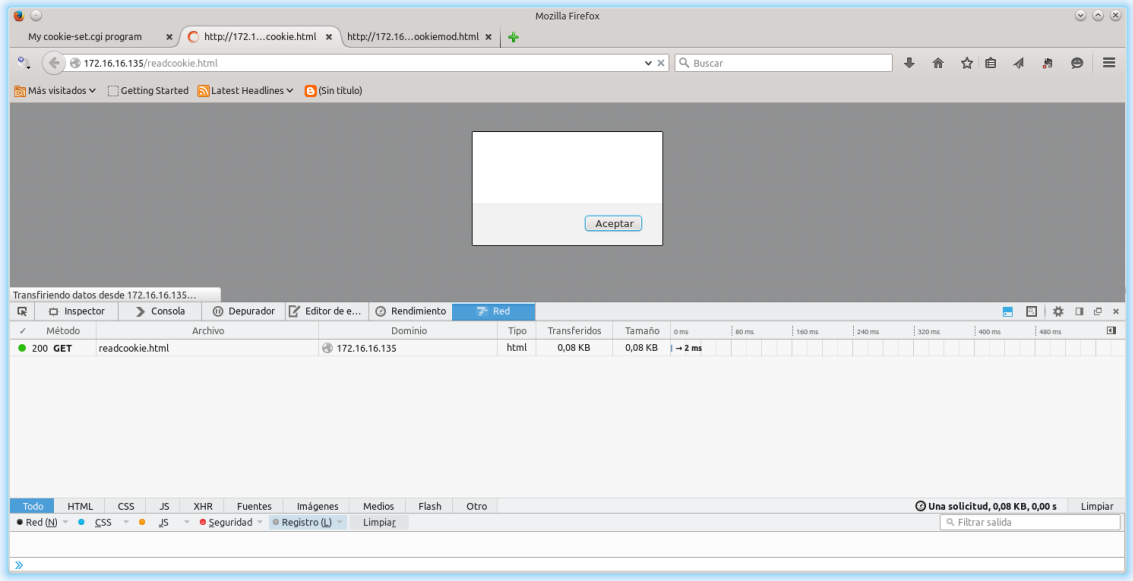
Figura 40. Creación de cookie con httpOnly.




Fuente: autor del proyecto.

- Cargar la página http://172.16.16.135/readcookie.html. Este html contiene un script en código javascript, el cual intenta leer la cookie almacenada. En la figura 41 puede verse que dicho script no logra hacer lectura de la cookie. El código del html de la página puede ser consultado en el anexo D.

Figura 41. Lectura de cookie httpOnly con javascript.



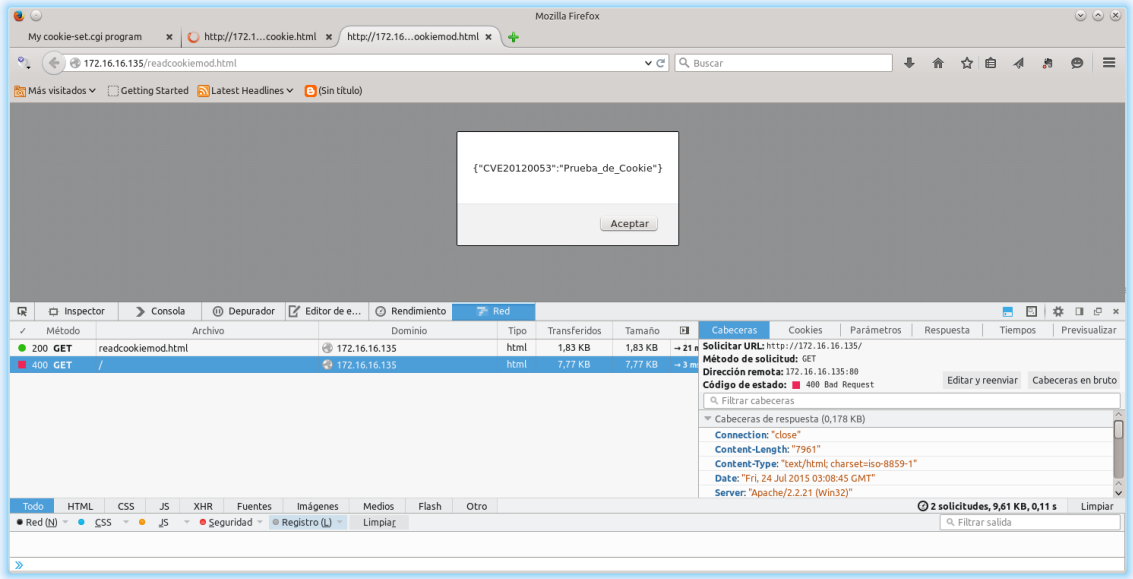

Fuente: autor del proyecto.

• Ejecutar el html del sitio http://172.16.16.135/readcookiemod.html. Este html contiene el código necesario para explotar la vulnerabilidad en cuestión. Como puede verse en la figura 42, la vulnerabilidad ha sido explotada exitosamente, de modo que se han obtenido los valores de la cookie httpOnly a través del exploit compuesto por código javascript. El código del html puede ser consultado en el anexo E.

• El siguiente paso consiste en repetir los 3 pasos anteriores, pero esta vez con el servidor HTTP Apache 1.3.17 activo, el cual es el objetivo de estudio. Una vez se haya realizado la activación de este servicio, y habiendo borrado tanto el historial como las cookies almacenadas localmente, se procede a ingresar nuevamente a ejecutar el script http://172.16.16.135/cgi-bin/setcookie.pl. El resultado es el mismo observado en la figura 40, el cual es la creación de la cookie.

• Luego se procede a cargar el html del sitio http://172.16.16.135/readcookie.html. Una vez más, el resultado es el mismo que cuando se ejecutó en el servidor HTTP Apache versión 2.2.21, el cual puede verse en la figura 41.

Figura 42. Lectura de cookie httpOnly con exploit en apache v2.2.21.

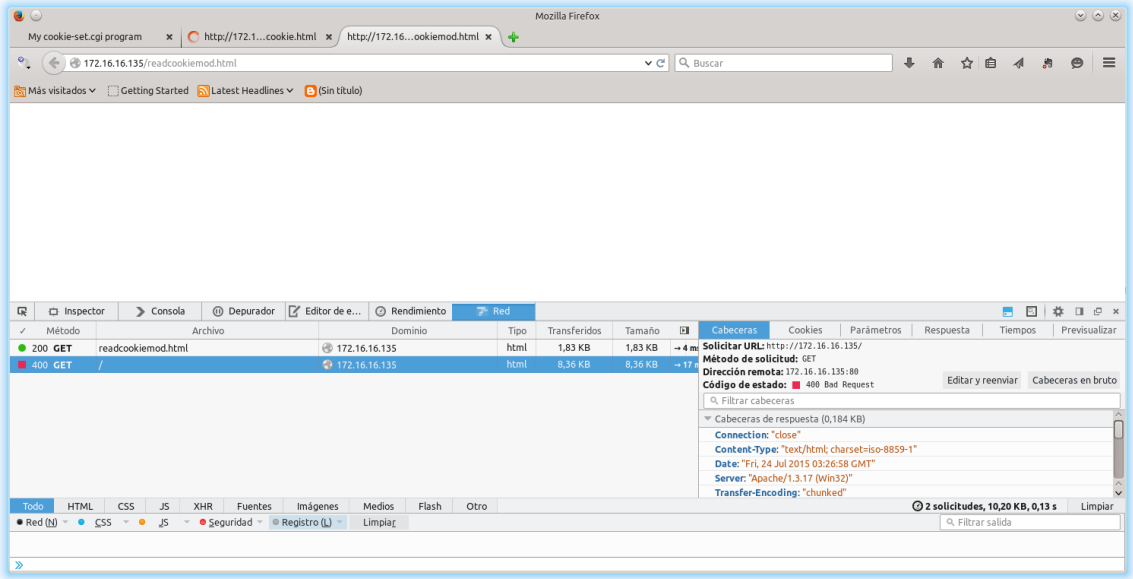



Fuente: autor del proyecto.

• Y por último, se ejecuta el html que contiene el código para explotar la vulnerabilidad http://172.16.16.135/readcookiemod.html. Puede notarse en la figura 43, que a diferencia de la versión 2.2.21, la versión 1.3.17 del servidor HTTP Apache no obtiene los valores de la cookie tal cual sucedió anteriormente.

Conforme las pruebas de laboratorio realizadas, se ha demostrado que el servidor HTTP Apache versión 1.3.17 no es explotable tal como hace referencia el CVE-2012-0053.

Figura 43. Lectura de cookie httpOnly con exploit en apache v1.3.17.




Fuente: autor del proyecto.

### 5.1.6 Comparativa funcional de un escáner de vulnerabilidades tradicional.

Un buen punto de partida para determinar la “calidad” de un producto, es comparándolo con otro producto de características comparables, que cubra la misma necesidad, y que sea altamente difundido y recomendado en el mercado. Para realizar esta comparativa, se ha elegido el conocido escáner de vulnerabilidades Nessus en su versión Home, el cual es idóneo para cumplir con el objetivo debido a las siguientes características:

- Es un escáner de vulnerabilidades conocido y con una gran trayectoria.
- Comparte características técnicas con OpenVAS, ya que este último se creó como una alternativa de código abierto cuando Nessus decidió convertirse en software privativo.
- La versión Home no tiene costo, por lo que no se incurren en costos adicionales para el estudio en mención.

El objetivo de esta comparativa es determinar de una manera rápida pero acertada, hasta donde puede ser útil el escáner de vulnerabilidad OpenVAS, en que situaciones podría arrojar buenos resultados, y sobre todo en cuáles no.

Las comparaciones a realizar entre OpenVAS y Nessus se evaluaran tomando en cuenta los siguientes parámetros:

- Tiempo que tarda en realizar el escaneo de vulnerabilidades.
- Vulnerabilidades halladas.
- Requerimientos de recursos de hardware del servidor.

**5.1.6.1 Vulnerabilidades halladas por los escáneres.** Como puede verse en el anexo F, Nessus además de clasificar algunas vulnerabilidades como *Info* (equivalentes con las vulnerabilidades marcadas como *Log* en OpenVAS), clasifica como *Critical* aquellas que tienen una puntuación CVSS de 10.0. Para facilitar la comparación entre los dos escáneres, las vulnerabilidades clasificadas como *Critical* se trataran como *High*, y se omiten las vulnerabilidades clasificadas como *Log* e *Info,* ya que estas no representan un riesgo evidente como sí lo son el resto.

El resumen de las vulnerabilidades halladas por Nessus puede verse en la tabla 9.

Tabla 9. Vulnerabilidades halladas por Nessus.

| Nombre | Amenaza |
|---|---|
| MS08-067: Microsoft Windows Server Service Crafted RPC Request Handling Remote Code Execution (958644) (uncredentialed check) | Critical (CVSS: 10.0) |
| MS09-001: Microsoft Windows SMB Vulnerabilities Remote Code Execution (958687) (uncredentialed check) | Critical (CVSS: 10.0) |
| Microsoft Windows XP Unsupported Installation Detection | Critical (CVSS: 10.0) |
| MS12-020: Vulnerabilities in Remote Desktop Could Allow Remote Code Execution (2671387) (uncredentialed check) | High (CVSS: 9.3) |
| Apache Chunked Encoding Remote Overflow | High (CVSS: 7.5) |
| Apache < 1.3.27 Multiple Vulnerabilities (DoS, XSS) | High (CVSS: 7.5) |
| Apache < 1.3.37 mod_rewrite LDAP Protocol URL Handling Overflow | High (CVSS: 7.5) |
| Unsupported Web Server Detection | High (CVSS: 7.5) |
| Apache < 1.3.29 Multiple Modules Local Overflow | High (CVSS: 7.2) |
| Apache < 1.3.28 Multiple Vulnerabilities (DoS, ID) | High (CVSS: 7.1) |
| Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness | Medium (CVSS: 5.1) |
| Apache < 1.3.31 / 2.0.49 Socket Connection Blocking Race Condition DoS | Medium (CVSS: 5.0) |
| SSL Version 2 and 3 Protocol Detection | Medium (CVSS: 5.0) |
| Microsoft Windows SMB NULL Session Authentication | Medium (CVSS: 5.0) |
| SMB Signing Required | Medium (CVSS: 5.0) |
| HTTP TRACE / TRACK Methods Allowed | Medium (CVSS: 4.3) |
| Web Server Expect Header XSS | Medium (CVSS: 4.3) |
| SSL Weak Cipher Suites Supported | Medium (CVSS: 4.3) |
| Apache < 1.3.41 Multiple Vulnerabilities (DoS, XSS) | Medium (CVSS: 4.3) |
| SSL Medium Strength Cipher Suites Supported | Medium (CVSS: 4.3) |
| Terminal Services Encryption Level is Medium or Low | Medium (CVSS: 4.3) |
| Apache HTTP Server httpOnly Cookie Information Disclosure | Medium (CVSS: 4.3) |
| SSL RC4 Cipher Suites Supported | Medium (CVSS: 4.3) |
| SSLv3 Padding Oracle On Downgraded Legacy Encryption Vulnerability (POODLE) | Medium (CVSS: 4.3) |
| Terminal Services Encryption Level is not FIPS-140 Compliant | Low (CVSS: 2.6) |
| SSL Anonymous Cipher Suites Supported | Low (CVSS: 2.6) |

Fuente: autor del proyecto.

La tabla 10 es un comparativo que muestra las cantidades de vulnerabilidades halladas tanto por OpenVAS como por Nessus. En esta tabla puede observarse que Nessus detecta más del doble de las vulnerabilidades halladas por OpenVAS.

Tabla 10. Cantidad de vulnerabilidades por clasificación de amenaza.

| Clasificación | OpenVAS | Nessus |
|---|---|---|
| High | 3 | 10 |
| Medium | 6 | 14 |
| Low | 3 | 2 |

Fuente: autor del proyecto.

De acuerdo a estos datos, podemos determinar que la eficacia de OpenVAS con respecto de Nessus es:

$$Eficacia_F = \frac{Vulnerabilidades\ halladas\ OpenVAS \times 100}{Vulnerabilidades\ halladas\ Nessus}$$

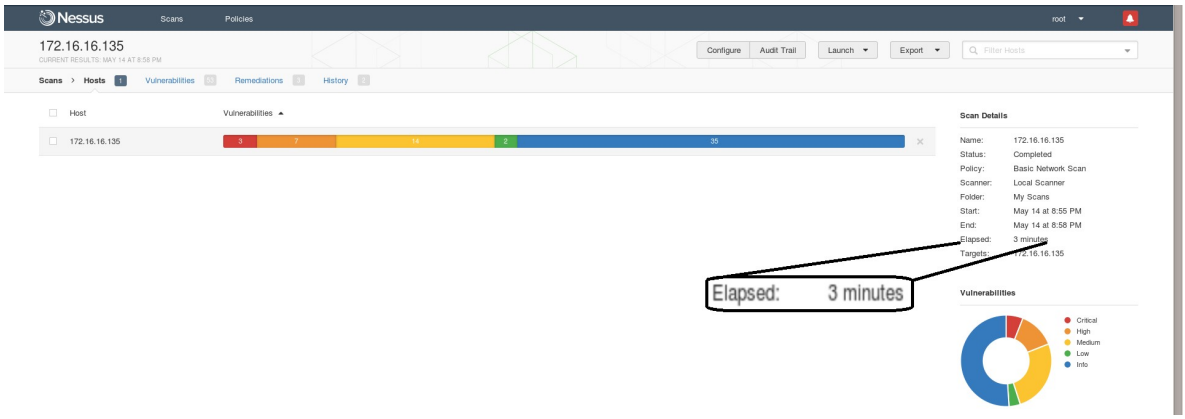
$$Eficacia_F = \frac{12\,V \times 100}{26\,V}$$

$$Eficacia_F = 46\,\%$$

**5.1.6.2 Tiempos de escaneo.** Como pudo determinarse anteriormente, el tiempo promedio de escaneo de OpenVAS integrado como plugin de Nagios es de 17 minutos y 4 segundos. A pesar que este tiempo corresponde a una integración de OpenVAS con Nagios, nos sirve como punto de comparación debido a que los retrasos provocados por el plugin son mínimos, y además la configuración del plugin esta optimizada para obtener unos resultados mayormente acertados con respecto al escaneo de evaluación inicial ejecutado durante la evaluación del producto.

En cuanto a Nessus, el escaneo de vulnerabilidades básico del host objetivo tomo 3 minutos, tal como se muestra en la figura 44.

Figura 44. Tiempo que tardo Nessus en escanear el host objetivo.




Fuente: autor del proyecto.

Ahora podemos determinar mediante las siguientes formulas. La eficiencia del tiempo por parte de OpenVAS con respecto de Nessus:

Eficiencia Nessus:

$$Eficiencia_N = \frac{Vulnerabilidades\ halladas}{Tiempo}$$

$$Eficiencia_N = \frac{26\,V}{3\,min}$$

$$Eficiencia_N = 8.66\,V/min$$

Eficiencia OpenVAS:

$$Eficiencia_O = \frac{Vulnerabilidades\ halladas}{Tiempo}$$

$$Eficiencia_O = \frac{12\,V}{17\,min}$$

$$Eficiencia_O = 0.7\,V/min$$

Finalmente tenemos que la eficiencia de OpenVAS con respecto de Nessus es:

$$Eficiencia_F = \frac{Eficiencia\,OpenVAS \times 100}{Eficiencia\,Nessus}$$

$$Eficiencia_F = \frac{0.7\,V/min \times 100}{8.66\,V/min}$$

$$Eficiencia_F = 8\,\%$$

**5.1.6.3 Recursos de hardware del servidor empleado.** Con el fin de simplificar la comparativa, los recursos de hardware destinados al servidor de Nessus son los mismos que los destinados al servidor OpenVAS. En la tabla 11 se detallan las características técnicas del servidor Nessus empleado.

Tabla 11. Características técnicas del servidor Nessus.

| Ítem | Detalle |
|---|---|
| Servidor: | Virtual – VMWare Workstation 10.0.3 |
| Sistema operativo: | Debian Linux 7.6 |
| Versión de Nagios: | Nagios core v3.0.6 |
| Versión de Nessus: | Nessus Home Edition v6.3.6 |
| Procesador: | Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. 2 Núcleos asignados. |
| Memoria: | 1 GB |
| Disco duro: | 15 GB |

Fuente: autor del proyecto.

# 6. CONCLUSIONES Y RECOMENDACIONES

Una motivación que dio inicio este proyecto, fue la de proporcionar al lector o interesado un método de monitoreo de las vulnerabilidades de bajo coste, fácilmente implementable y con una efectividad aceptable, buscando como eje central aprovechar los beneficios del sistema de monitoreo de redes Nagios. Dicha motivación fue impulsada por el previo conocimiento de la existencia del módulo Vulscan.nse para nmap por parte del autor, quien en su momento tenía una buena expectativa acerca de su funcionamiento y desempeño.

En la búsqueda de las alternativas actuales de código abierto para el escaneo de vulnerabilidades, que por cierto no fue sencilla, a parte del módulo Vulscan.nse se encuentran los sistemas OpenVAS y SARA. Es importante mencionar que el abanico de alternativas de código abierto en este segmento de software es muy limitado actualmente, ya que muchos escáneres de vulnerabilidades que iniciaron como software libre, terminaron convirtiéndose en software privativo como el ya mencionado caso de Nessus.

Derivado del estudio realizado en este trabajo, puede concluirse de los escáneres de vulnerabilidades evaluados lo siguiente:

- SARA es un sistema abandonado el cual en la actualidad no tiene ninguna aplicación funcional.
- El módulo Vulscan.nse es un proyecto sencillo, interesante, pero muy joven para ser una alternativa seria a un escáner de vulnerabilidades. Desafortunadamente, durante el desarrollo del proyecto dicho módulo ha cumplido 2 años sin recibir actualización alguna, lo que hace pensar que éste se convertirá en un proyecto abandonado justo antes de ser útil para la comunidad.
- OpenVAS sin embargo, es la alternativa de código abierto más fuerte que existe actualmente en el mercado. A pesar de la eficiencia que evidentemente puede mejorarse en futuras versiones, y su eficacia medianamente aceptable, puede llegar ser útil en ciertos entornos donde se cuentan con escasos recursos económicos para la inversión en software, y se desea adicionar una capa de seguridad inexistente.

De la integración de OpenVAS con Nagios, se puede decir que realmente no es compleja debido a la flexibilidad de la interfaz CLI de OpenVAS, lo cual permite adaptarse perfectamente a cualquier shell script. Además, la facilidad de OpenVAS

de permitir generar informes de escaneos en formato texto, permite realizar filtros avanzados para extraer la información de interés y posteriormente ser mostrada en la salida del plugin de Nagios.

Cabe mencionar que es una desgracia el tiempo que le toma a OpenVAS en realizar el análisis completo del host objetivo, ya que esto obliga a implementar chequeos pasivos en Nagios, los cuales no son los más convenientes para estos casos. A pesar de ello, dicha implementación es funcional y aceptable para realizar el seguimiento de un host objetivo tal cual se ha mostrado en el estudio.

Para llegar a monitorear más objetivos simultáneamente, lo más apropiado es usar un hardware de mayor capacidad tipo servidor, ya que los sistemas empleados en este estudio fueron entornos de pruebas virtualizados con bajos recursos de hardware. Se debe tener en cuenta también que el uso de un hardware más potente posiblemente disminuya el tiempo promedio de escaneo empleado por OpenVAS. Cabe aclarar que con las pruebas realizadas en este estudio, es prematuro crear una fórmula que permita dimensionar el hardware que se requiere para monitorear una cantidad determinada de hosts. Para llegar a determinar una formula aproximada, se requiere de un entorno de pruebas más robusto el cual sobrepasa el alcance inicial del proyecto.

Como ya se mencionó anteriormente, el plugin check_openvas fue desarrollado en su totalidad en shell script. Básicamente cualquier información obtenida en el reporte generado por OpenVAS puede incluirse en la salida del plugin, sin embargo, en la práctica se suele incluir sólo información relevante. Para este caso, la salida del plugin retorna la cantidad de vulnerabilidades altas, medias, bajas e informativas. La razón de ello, es que dichos valores son medibles y cuantificables, lo cual facilita la generación de gráficas mediante la implementación de un software como lo es el generador de gráficos de red conocido como Cacti, el cual es ampliamente implementado con el sistema Nagios.

Si el lector se ha fijado en algunas figuras donde se muestra la salida del plugin, notará que además de retornar las cantidades de vulnerabilidades discriminadas de acuerdo a su clasificación, el plugin retorna también una dirección URL en donde es posible descargar y visualizar el reporte en formato texto generado por OpenVAS. Esto con el fin de facilitar la gestión de las vulnerabilidades, de modo que el plugin no solamente se limite a informar cantidades que en si son poco útiles, sino que también se pueda consultar el detalle de dichas vulnerabilidades.

La implementación de OpenVAS y Nagios realizada durante este estudio, contempla la instalación de los dos aplicativos en el mismo servidor. Sin embargo, esto no fue debido a una limitación del plugin como tal; el plugin puede configurarse de modo que se ejecute desde un servidor Nagios que puede ser o no diferente al servidor donde se ejecuta OpenVAS. En caso que se quiera separar OpenVAS, el plugin puede ser ajustado para conectarse a un servidor OpenVAS remoto pasándole las opciones “--host” y “--port” al comando “omp” para indicarle al plugin la IP y el puerto en que está escuchando el servidor de OpenVAS. Esto permite dar escalabilidad a la solución planteada, al permitir usar un servidor exclusivo para OpenVAS totalmente ajeno e independiente al servidor Nagios.

Para finalizar, es importante concluir que el objetivo del proyecto se ha cumplido satisfactoriamente, ya que se ha proporcionado al lector un método y un conjunto de herramientas de código abierto con el cual puede monitorear las vulnerabilidades de uno o varios host objetivo utilizando el sistema Nagios. La solución final propuesta aquí se basa en el uso del escáner de vulnerabilidades OpenVAS, integrado al sistema Nagios por medio del plugin producto de este proyecto denominado check_openvas, cuyo código es abierto y se encuentra publicado en el anexo G. No obstante, es importante mencionar que esta es una solución que no tiene costos relativos al software, y se recomienda su implementación siempre y cuando:

- La compañía o entidad que implementará dicha solución no posea recursos para la implementación de un software escáner de vulnerabilidades comercial y con su debido soporte.
- Este dispuesto a correr el riesgo de su uso, conociendo previamente la eficacia de OpenVAS con respecto a Nessus.

Si se atiende a las recomendaciones dadas, la solución propuesta puede ser de gran utilidad para el lector y las empresas que consideren su implementación.

# 7. BIBLIOGRAFÍA

Advanced Research Corporation. [online]. [citado el 17 de Agosto de 2014] Disponible en internet: <http://www-arc.com/sara/>.

Computec.ch. [online]. [citado el 24 de Agosto de 2014] Disponible en internet: <http://www.computec.ch/projekte/vulscan/?>.

Nagios Enterprises, LLC. About Nagios Core [online]. [citado en 10 febrero de 2014] Disponible en internet: <http://nagios.sourceforge.net/docs/nagioscore/3/en/about.html#whatis>.

OpenVAS. [online]. [citado el 09 de Agosto de 2014] Disponible en internet: <http://www.openvas.org/>.

ANEXO A. Reporte de OpenVAS para la evaluación inicial.

# Scan Report

June 10, 2015

**Summary**

This document reports on the results of an automatic security scan. The scan started at Wed May 27 02:44:25 2015 UTC and ended at Wed May 27 02:55:43 2015 UTC. The report first summarises the results found. Then, for each host, the report describes every issue found. Please consider the advice given in each description, in order to rectify the issue.

## Contents

# 1 Result Overview

| Host | Most Severe Result(s) | High | Medium | Low | Log | False Positives |
|---|---|---|---|---|---|---|
| 172.16.16.135 (CAMILO-028F067D) | Severity: High | 3 | 4 | 6 | 28 | 0 |
| Total: 1 | | 3 | 4 | 6 | 28 | 0 |

Vendor security updates are not trusted.
Overrides are on. When a result has an override, this report uses the threat of the override.
Notes are included in the report.
This report might not show details of all issues that were found.
It only lists hosts that produced issues.
Issues with the threat level "Debug" are not shown.

This report contains all 41 results selected by the filtering described above. Before filtering there were 41 results.

# 2 Results per Host

## 2.1 172.16.16.135

| | |
|---|---|
| Host scan start | Wed May 27 02:44:34 2015 UTC |
| Host scan end | Wed May 27 02:55:43 2015 UTC |

| Service (Port) | Threat Level |
|---|---|
| http (80/tcp) | High |
| microsoft-ds (445/tcp) | High |
| ms-wbt-server (3389/tcp) | High |
| http (80/tcp) | Medium |
| ms-wbt-server (3389/tcp) | Low |
| general/SMBClient | Low |
| ntp (123/udp) | Low |
| http (80/tcp) | Log |
| microsoft-ds (445/tcp) | Log |
| ms-wbt-server (3389/tcp) | Log |
| epmap (135/tcp) | Log |
| general/CPE-T | Log |
| general/HOST-T | Log |
| general/icmp | Log |
| general/tcp | Log |
| netbios-ns (137/udp) | Log |
| netbios-ssn (139/tcp) | Log |
| netsaint (5666/tcp) | Log |

### 2.1.1 High http (80/tcp)

High (CVSS: 5.8)
NVT: http TRACE XSS attack

```
Solution:
Add the following lines for each virtual host in your configuration file :
    RewriteEngine on
    RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK)
    RewriteRule .* - [F]
See also http://httpd.apache.org/docs/current/de/mod/core.html#traceenable
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.11213

**References**

```
CVE: CVE-2004-2320, CVE-2003-1567
BID:9506, 9561, 11604
Other:
  URL:http://www.kb.cert.org/vuls/id/867593
```

[ return to 172.16.16.135 ]

### 2.1.2 High microsoft-ds (445/tcp)

High (CVSS: 10.0)
NVT: Microsoft Windows SMB Server NTLM Multiple Vulnerabilities (971468)

```
Summary:
This host is missing a critical security update according to
Microsoft Bulletin MS10-012.
Vulnerability Insight:
- An input validation error exists while processing SMB requests and can
  be exploited to cause a buffer overflow via a specially crafted SMB packet.
- An error exists in the SMB implementation while parsing SMB packets during
  the Negotiate phase causing memory corruption via a specially crafted SMB
  packet.
- NULL pointer dereference error exists in SMB while verifying the 'share'
  and 'servername' fields in SMB packets causing denial of service.
- A lack of cryptographic entropy when the SMB server generates challenges
  during SMB NTLM authentication and can be exploited to bypass the
  authentication mechanism.
Impact:
Successful exploitation will allow remote attackers to execute arbitrary
code or cause a denial of service or bypass the authentication mechanism
via brute force technique.
Impact Level: System/Application
```

... continues on next page ...

... continued from previous page ...

Affected Software/OS:
Microsoft Windows 7
Microsoft Windows 2000 Service Pack and prior
Microsoft Windows XP Service Pack 3 and prior
Microsoft Windows Vista Service Pack 2 and prior
Microsoft Windows Server 2003 Service Pack 2 and prior
Microsoft Windows Server 2008 Service Pack 2 and prior
Solution:
Run Windows Update and update the listed hotfixes or download and
update mentioned hotfixes in the advisory from the below link,
http://www.microsoft.com/technet/security/bulletin/ms10-012.mspx

OID of test routine: 1.3.6.1.4.1.25623.1.0.902269

**References**

CVE: CVE-2010-0020, CVE-2010-0021, CVE-2010-0022, CVE-2010-0231
Other:
URL:http://secunia.com/advisories/38510/
URL:http://support.microsoft.com/kb/971468
URL:http://www.vupen.com/english/advisories/2010/0345
URL:http://www.microsoft.com/technet/security/bulletin/ms10-012.mspx

[ return to 172.16.16.135 ]

### 2.1.3 High ms-wbt-server (3389/tcp)

High (CVSS: 9.3)
NVT: Microsoft Remote Desktop Protocol Remote Code Execution Vulnerabilities (2671387)

Summary:
This host is missing a critical security update according to
Microsoft Bulletin MS12-020.
Vulnerability Insight:
The flaws are due to the way Remote Desktop Protocol accesses an
object in memory that has been improperly initialized or has been deleted
and the way RDP service processes the packets.
Impact:
Successful exploitation could allow remote attackers to execute arbitrary
code as the logged-on user or cause a denial of service condition.
Impact Level: System/Application
Affected Software/OS:
Microsoft Windows 7 Service Pack 1 and prior
Microsoft Windows XP Service Pack 3 and prior

... continues on next page ...

... continued from previous page ...

```
Microsoft Windows 2K3 Service Pack 2 and prior
Microsoft Windows Vista Service Pack 2 and prior
Microsoft Windows Server 2008 Service Pack 2 and prior
Solution:
Run Windows Update and update the listed hotfixes or download and
update mentioned hotfixes in the advisory from the below link,
http://technet.microsoft.com/en-us/security/bulletin/ms12-020
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.902818

**References**

```
CVE: CVE-2012-0002, CVE-2012-0152
BID:52353, 52354
Other:
  URL:http://blog.binaryninjas.org/?p=58
   URL:http://secunia.com/advisories/48395
   URL:http://support.microsoft.com/kb/2671387
   URL:http://www.securitytracker.com/id/1026790
   URL:http://technet.microsoft.com/en-us/security/bulletin/ms12-020
```

[ return to 172.16.16.135 ]

### 2.1.4 Medium http (80/tcp)

Medium (CVSS: 5.0)
NVT: Check for Apache Multiple / vulnerability

```
It is possible to list a directories contents by appending multiple /'s
in the HTTP GET command, this is only
a vulnerability on Apache/Win32 based webservers. 197 slashes will cause the dir
↪ectory contents to be listed
Solution: Upgrade to the most recent version of Apache at www.apache.org
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.10440

**References**

```
CVE: CVE-2000-0505
BID:1284
```

Medium (CVSS: 5.0)
NVT: Apache Connection Blocking Denial of Service

Summary:
The remote web server appears to be running a version of
Apache that is less that 2.0.49 or 1.3.31.
These versions are vulnerable to a denial of service attack where a remote
attacker can block new connections to the server by connecting to a listening
socket on a rarely accessed port.
Solution:
Upgrade to Apache 2.0.49 or 1.3.31.

OID of test routine: 1.3.6.1.4.1.25623.1.0.12280

**References**
CVE: CVE-2004-0174
BID:9921

Medium (CVSS: 4.3)
NVT: Apache Web Server ETag Header Information Disclosure Weakness

Information that was gathered:
Inode: 0
Size: 1354

OID of test routine: 1.3.6.1.4.1.25623.1.0.103122

**References**
CVE: CVE-2003-1418
BID:6939
Other:
URL:https://www.securityfocus.com/bid/6939
URL:http://httpd.apache.org/docs/mod/core.html#fileetag
URL:http://www.openbsd.org/errata32.html
URL:http://support.novell.com/docs/Tids/Solutions/10090670.html

Medium (CVSS: 4.3)
NVT: Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability

Summary:
. . . continues on next page . . .

... continued from previous page ...

```
This host is running Apache HTTP Server and is prone to cookie
information disclosure vulnerability.
Vulnerability Insight:
The flaw is due to an error within the default error response for
status code 400 when no custom ErrorDocument is configured, which can be
exploited to expose 'httpOnly' cookies.
Impact:
Successful exploitation will allow attackers to obtain sensitive information
that may aid in further attacks.
Impact Level: Application
Affected Software/OS:
Apache HTTP Server versions 2.2.0 through 2.2.21
Solution:
Upgrade to Apache HTTP Server version 2.2.22 or later,
For updates refer to http://httpd.apache.org/
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.902830

**References**

```
CVE: CVE-2012-0053
BID:51706
Other:
  URL:http://osvdb.org/78556
   URL:http://secunia.com/advisories/47779
   URL:http://www.exploit-db.com/exploits/18442
   URL:http://rhn.redhat.com/errata/RHSA-2012-0128.html
   URL:http://httpd.apache.org/security/vulnerabilities_22.html
   URL:http://svn.apache.org/viewvc?view=revision&amp;revision=1235454
   URL:http://lists.opensuse.org/opensuse-security-announce/2012-02/msg00026.htm
↪l
```

[ return to 172.16.16.135 ]

### 2.1.5 Low ms-wbt-server (3389/tcp)

Low (CVSS: 0.0)
NVT: Microsoft Remote Desktop Protocol Detection

```
Summary:
The Microsoft Remote Desktop Protocol (RDP) is running at this host. Remote
Desktop Services, formerly known as Terminal Services, is one of the components
of Microsoft Windows (both server and client versions) that allows a user to
access applications and data on a remote computer over a network.
```

... continues on next page ...

... continued from previous page ...

OID of test routine: 1.3.6.1.4.1.25623.1.0.100062

[ return to 172.16.16.135 ]

### 2.1.6 Low general/SMBClient

Low (CVSS: 0.0)
NVT: SMB Test

```
OS Version = WINDOWS 5.1
Domain = WORKGROUP
SMB Serverversion = WINDOWS 2000 LAN MANAGER
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.90011

Low (CVSS: 0.0)
NVT: SMB Test

```
OS Version = WINDOWS 5.1
Domain = WORKGROUP
SMB Serverversion = Windows 2000 LAN Manager
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.90011

Low (CVSS: 0.0)
NVT: SMB Test

```
OS Version = Windows 5.1
Domain = WORKGROUP
SMB Serverversion = WINDOWS 2000 LAN MANAGER
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.90011

Low (CVSS: 0.0)
NVT: SMB Test

```
OS Version = Windows 5.1
Domain = WORKGROUP
SMB Serverversion = Windows 2000 LAN Manager
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.90011

[ return to 172.16.16.135 ]

### 2.1.7 Low ntp (123/udp)

Low (CVSS: 0.0)
NVT: NTP read variables

```
Summary:
 A NTP (Network Time Protocol) server is listening on this port.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.10884

[ return to 172.16.16.135 ]

### 2.1.8 Log http (80/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

Log (CVSS: 0.0)
NVT: HTTP Server type and version

```
The remote web server type is :
Apache/1.3.17 (Win32)
Solution : You can set the directive 'ServerTokens Prod' to limit
```

... continues on next page ...

... continued from previous page ...

the information emanating from the server in its response headers.

OID of test routine: 1.3.6.1.4.1.25623.1.0.10107

Log (CVSS: 0.0)
NVT: Services

A web server is running on this port

OID of test routine: 1.3.6.1.4.1.25623.1.0.10330

Log (CVSS: 0.0)
NVT: Web mirroring

The following CGI have been discovered :
Syntax : cginame (arguments [default value])
/manual/howto/ (D [A] M [A] N [A] D=D [] S [A] )
Directory index found at /manual/howto/

OID of test routine: 1.3.6.1.4.1.25623.1.0.10662

Log (CVSS: 0.0)
NVT: Directory Scanner

The following directories were discovered:
/cgi-bin, /icons, /manual
While this is not, in and of itself, a bug, you should manually inspect
these directories to ensure that they are in compliance with company
security standards

OID of test routine: 1.3.6.1.4.1.25623.1.0.11032

**References**

Other:
OWASP:OWASP-CM-006

Log (CVSS: 0.0)
NVT: wapiti (NASL wrapper)

```
wapiti could not be found in your system path.
OpenVAS was unable to execute wapiti and to perform the scan you
requested.
Please make sure that wapiti is installed and that wapiti is
available in the PATH variable defined for your environment.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.80110

Log (CVSS: 0.0)
NVT: Apache Web ServerVersion Detection

```
Detected Apache
Version: 1.3.17
Location: 80/tcp
CPE: cpe:/a:apache:http_server:1.3.17
Concluded from version identification result:
Server: Apache/1.3.17
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.900498

[ return to 172.16.16.135 ]

### 2.1.9 Log microsoft-ds (445/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

Log (CVSS: 0.0)
NVT: SMB NativeLanMan

```
Summary:
It is possible to extract OS, domain and SMB server information
```

...continues on next page ...

... continued from previous page ...

```
from the Session Setup AndX Response packet which is generated
during NTLM authentication.Detected SMB workgroup: WORKGROUP
Detected SMB server: Windows 2000 LAN Manager
Detected OS: Windows 5.1
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.102011

Log (CVSS: 0.0)
NVT: SMB on port 445

```
A CIFS server is running on this port
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.11011

[ return to 172.16.16.135 ]

#### 2.1.10 Log ms-wbt-server (3389/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

Log (CVSS: 0.0)
NVT: Identify unknown services with nmap

```
Nmap service detection result for this port: ms-wbt-server
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.66286

[ return to 172.16.16.135 ]

#### 2.1.11 Log epmap (135/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

[ return to 172.16.16.135 ]

### 2.1.12 Log general/CPE-T

Log (CVSS: 0.0)
NVT: CPE Inventory

```
172.16.16.135|cpe:/a:apache:http_server:1.3.17
172.16.16.135|cpe:/o:microsoft:windows
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.810002

[ return to 172.16.16.135 ]

### 2.1.13 Log general/HOST-T

Log (CVSS: 0.0)
NVT: Host Summary

```
traceroute:172.16.16.133,172.16.16.135
TCP ports:5666,445,135,139,3389,80
UDP ports:
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.810003

[ return to 172.16.16.135 ]

### 2.1.14 Log general/icmp

Log (CVSS: 0.0)
NVT: ICMP Timestamp Detection

```
Summary:
The remote host responded to an ICMP timestamp request. The Timestamp Reply is
an ICMP message which replies to a Timestamp message. It consists of the
originating timestamp sent by the sender of the Timestamp as well as a receive
timestamp and a transmit timestamp. This information could theoretically be used
to exploit weak time-based random number generators in other services.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.103190

**References**

```
CVE: CVE-1999-0524
Other:
  URL:http://www.ietf.org/rfc/rfc0792.txt
```

[ return to 172.16.16.135 ]

### 2.1.15 Log general/tcp

Log (CVSS: 0.0)
NVT: OS fingerprinting

```
ICMP based OS fingerprint results: (100% confidence)
Microsoft Windows
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.102002

**References**

```
Other:
  URL:http://www.phrack.org/issues.html?issue=57&amp;id=7#article
```

Log (CVSS: 0.0)
NVT: DIRB (NASL wrapper)

```
DIRB could not be found in your system path.
OpenVAS was unable to execute DIRB and to perform the scan you
requested.
Please make sure that DIRB is installed and is
```

... continues on next page ...

... continued from previous page ...

```
available in the PATH variable defined for your environment.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.103079

Log (CVSS: 0.0)
NVT: Checks for open udp ports

```
Open UDP ports: [None found]
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.103978

Log (CVSS: 0.0)
NVT: arachni (NASL wrapper)

```
Arachni could not be found in your system path.
OpenVAS was unable to execute Arachni and to perform the scan you
requested.
Please make sure that Arachni is installed and that arachni is
available in the PATH variable defined for your environment.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.110001

Log (CVSS: 0.0)
NVT: Nikto (NASL wrapper)

```
Nikto could not be found in your system path.
OpenVAS was unable to execute Nikto and to perform the scan you
requested.
Please make sure that Nikto is installed and that nikto.pl or nikto is
available in the PATH variable defined for your environment.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.14260

... continues on next page ...

... continued from previous page ...

Log (CVSS: 0.0)
NVT: Traceroute

```
Here is the route from 172.16.16.133 to 172.16.16.135:
172.16.16.133
172.16.16.135
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.51662

Log (CVSS: 0.0)
NVT: Microsoft SMB Signing Disabled

```
SMB signing is disabled on this host
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.802726

Log (CVSS: 0.0)
NVT: Checks for open tcp ports

```
Open TCP ports: 5666, 445, 135, 139, 3389, 80
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.900239

[ return to 172.16.16.135 ]

#### 2.1.16 Log netbios-ns (137/udp)

Log (CVSS: 0.0)
NVT: Using NetBIOS to retrieve information from a Windows host

```
The following 4 NetBIOS names have been gathered :
 CAMILO-028F067D
 WORKGROUP       = Workgroup / Domain name
 CAMILO-028F067D = This is the computer name
 WORKGROUP       = Workgroup / Domain name (part of the Browser elections)
The remote host has the following MAC address on its adapter :
   00:0c:29:90:b6:72
```

... continues on next page ...

... continued from previous page ...

```
If you do not want to allow everyone to find the NetBios name
of your computer, you should filter incoming traffic to this port.
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.10150

[ return to 172.16.16.135 ]

### 2.1.17 Log netbios-ssn (139/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

Log (CVSS: 0.0)
NVT: SMB on port 445

```
An SMB server is running on this port
```

OID of test routine: 1.3.6.1.4.1.25623.1.0.11011

[ return to 172.16.16.135 ]

### 2.1.18 Log netsaint (5666/tcp)

Log
NVT:

```
Open port.
```

OID of test routine: 0

[ return to 172.16.16.135 ]

This file was automatically generated.

Fuente: autor del proyecto.

ANEXO B. Primera página del reporte de Vulscan.nse para la evaluación inicial.

```
Starting Nmap 6.00 ( http://nmap.org ) at 2015-06-20 09:44 COT
Nmap scan report for 172.16.16.135
Host is up (0.00052s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE      VERSION
80/tcp   open  http         Apache httpd 1.3.17 ((Win32))
| vulscan: scip VulDB - http://www.scip.ch/en/?vuldb:
| [19654] Apache HTTP Server 1.3.19 auf HP Secure OS HTTP Request Handler unbekannte Schwachstelle
| [17637] Apache HTTP Server bis 1.3.19 auf Madrake Linux Directory Information Disclosure
| [16684] Apache HTTP Server bis 1.3.19 auf Win/OS2 HTTP Request Handler NULL Pointer Dereference Denial of Service
| [16548] Apache HTTP Server 1.3.19 mod_negotiation/mod_dir/mod_autoindex index.html Directory Information Disclosure
| [16534] Apache HTTP Server 1.3.14/2.0a9 htpasswd/htdigest /tmp Symlink erweiterte Rechte
| [16124] Apache HTTP Server bis 1.3.12 mod_rewrite RewriteRule Regular Expression erweiterte Rechte
| [16000] Apache HTTP Server 1.3.12 auf SuSE Linux WebDAV PROPFIND Directory Information Disclosure
| [15999] Apache HTTP Server 1.3.12 auf SuSE Linux /cgi-bin-sdb/ Source Information Disclosure
| [15298] Apache httpd bis 1.3.11 printenv.pl ap_send_error_response Cross Site Scripting
| [14792] Apache httpd bis 1.3.10 mod_rewrite/mod_vhost_alias erweiterte Rechte
| [14204] Apache httpd 1.3.1 MIME Header Handler Sioux Denial of Service
| [20310] Apache HTTP Server bis 1.3.24/2.0.45 Escape Character Handler erweiterte Rechte
| [19745] Apache HTTP Server bis 1.3.23 Log File Handler hostname Spoofing
| [19671] Apache HTTP Server 1.3.20 auf Windows /php/ erweiterte Rechte
| [19303] Apache HTTP Server 1.3.26/1.3.27 htdigest user Puffer\xC3\xBCberlauf
| [19128] Apache HTTP Server bis 1.3.27 Temp File Handler Symlink erweiterte Rechte
| [19052] Apache HTTP Server bis 1.3.26/2.0.42 ApacheBench ab.c Puffer\xC3\xBCberlauf
| [19051] Apache HTTP Server bis 1.3.26/2.0.42 Error Page Host Cross Site Scripting
| [19050] Apache HTTP Server bis 1.3.26 Shared Memory Scoreboard Denial of Service
| [18354] Apache HTTP Server bis 1.3.24/2.0.36 Chunked Encoding Transfer Handler Puffer\xC3\xBCberlauf
| [17988] Apache HTTP Server bis 1.3.23/2.0.34-beta auf Win32 erweiterte Rechte
| [17853] Apache HTTP Server bis 1.3.20 mod_usertrack schwache Authentisierung
| [17585] Apache HTTP Server 1.3.20 split-logfile Host erweiterte Rechte
| [17446] Apache HTTP Server bis 1.3.20 MultiviewHandler QUERY_STRING Directory Information Disclosure
| [17308] Apache HTTP Server bis 1.3.20 mod_rewrite erweiterte Rechte
| [16452] Apache HTTP Server bis 1.3.6 Directory Traversal
| [15810] Apache httpd 1.3.9/1.3.11/1.3.12 mod_vhost_alias /cgi-bin Source Information Disclosure
| [15608] Apache httpd bis 1.3.6.2 auf Windows Directory Information Disclosure
| [965] Apache httpd bis 1.3.32-r1 mod_include get_tag() Denial of Service
| [940] Apache httpd bis 1.3.31 mod_include get_tag() Puffer\xC3\xBCberlauf
| [706] Apache httpd bis 1.3.32 mod_proxy Content-Length Puffer\xC3\xBCberlauf
| [673] Apache httpd 1.3.x/2.0.x mod_ssl ssl_util_uuencode_binary() Puffer\xC3\xBCberlauf
| [638] Apache httpd bis 1.3.29 Secure Hash Handler Eingabeung\xC3\xBCltigkeit
| [185] Apache httpd bis 1.3.27 auf Win32/OS2 rotatelogs Denial of Service
|
| MITRE CVE - http://cve.mitre.org:
| [CVE-2012-1007] Multiple cross-site scripting (XSS) vulnerabilities in Apache Struts 1.3.10 allow remote attackers to inject arbitrary web script or HTML via (1) the name parameter to struts-examples/upload/upload-submit.do, or the message parameter to (2) struts-cookbook/processSimple.do or (3) struts-cookbook/processDyna.do.
| [CVE-2011-4449] actions/files/files.php in WikkaWiki 1.3.1 and 1.3.2, when INTRANET_MODE is enabled, supports file uploads for file extensions that are typically absent from an Apache HTTP Server TypesConfig file, which makes it easier for remote attackers to execute arbitrary PHP code by placing this code in a file whose name has multiple extensions, as demonstrated by a (1) .mm or (2) .vpp file.
| [CVE-2010-4408] Apache Archiva 1.0 through 1.0.3, 1.1 through 1.1.4, 1.2 through 1.2.2, and 1.3 through 1.3.1 does not require entry of the administrator's password at the time of modifying a user account, which makes it easier for context-dependent attackers to gain privileges by leveraging a (1) unattended workstation or (2) cross-site request forgery (CSRF) vulnerability, a related issue to CVE-2010-3449.
| [CVE-2010-3449] Cross-site request forgery (CSRF) vulnerability in Redback before 1.2.4, as used in Apache Archiva 1.0 through 1.0.3, 1.1 through 1.1.4, 1.2 through 1.2.2, and 1.3 through 1.3.1
| [CVE-2010-1623] Memory leak in the apr_brigade_split_line function in buckets/apr_brigade.c in the Apache Portable Runtime Utility library (aka APR-util) before 1.3.10, as used in the mod_reqtimeout module in the Apache HTTP Server and other software, allows remote attackers to cause a denial of service (memory consumption) via unspecified vectors related to the destruction of an APR bucket.
| [CVE-2007-2025] Unrestricted file upload vulnerability in the UpLoad feature (lib/plugin/UpLoad.php) in PhpWiki 1.3.11p1 allows remote attackers to upload arbitrary PHP files with a double extension, as demonstrated by .php.3, which is interpreted by Apache as being a valid PHP file.
```

Fuente: autor del proyecto.

ANEXO C. Código del script setcookie.pl.

```
#!/perl/bin/perl

use CGI;
$query = new CGI;

$cookie = $query->cookie(-name=>'CVE20120053',
»       »       »       -value=>'Prueba_de_Cookie',
»       »       »       -expires=>'+4h',
»       »       »       -httponly => 1,
»       »       »       -path=>'/');
print $query->header(-cookie=>$cookie);
print $query->start_html('My cookie-set.cgi program');
print $query->h3('The cookie has been set');
print $query->end_html;
```

Fuente: adaptada de http://alvinalexander.com/perl/edu/articles/pl010012.

ANEXO D. Código del archivo readcookie.html.

```
<html>
<body>
<script>
alert(document.cookie);
</script>
</body>
</html>
```

Fuente: adaptada de https://github.com/jonathansp/CVE20120053Demo/blob/master/www/readcookie.html.

ANEXO E. Código del archivo readcookiemod.html.

```
<html>
<body>
<script>
// Source: https://gist.github.com/1955a1c28324d4724b7b/7fe51f2a66c1d4a40a736540b3ad3fde02b7fb08
// Most browsers limit cookies to 4k characters, so we need multiple
function setCookies (good) {
    // Construct string for cookie value
    var str = "";
    for (var i=0; i< 819; i++) {
        str += "x";
    }
    // Set cookies
    for (i = 0; i < 10; i++) {
        // Expire evil cookie
        if (good) {
            var cookie = "xss"+i+"=;expires="+new Date(+new Date()-1).toUTCString()+"; path=/;";
        }
        // Set evil cookie
        else {
            var cookie = "xss"+i+"="+str+";path=/";
        }
        document.cookie = cookie;
    }
}

function makeRequest() {
    setCookies();

    function parseCookies () {
        var cookie_dict = {};
        // Only react on 400 status
        if (xhr.readyState === 4 && xhr.status === 400) {
            // Replace newlines and match <pre> content
            var content = xhr.responseText.replace(/\r|\n/g,'').match(/<pre>(.+)<\/pre>/);
            if (content.length) {
                // Remove Cookie: prefix
                content = content[1].replace("Cookie: ", "");
                var cookies = content.replace(/xss\d=x+;?/g, '').split(/;/g);
                // Add cookies to object
                for (var i=0; i<cookies.length; i++) {
                    var s_c = cookies[i].split('=',2);
                    cookie_dict[s_c[0]] = s_c[1];
                }
            }
            // Unset malicious cookies
            setCookies(true);
            alert(JSON.stringify(cookie_dict));
        }
    }
    // Make XHR request
    var xhr = new XMLHttpRequest();
    xhr.onreadystatechange = parseCookies;
    xhr.open("GET", "/", true);
    xhr.send(null);
}

makeRequest();
</script>
</body>
</html>
```

Fuente: adaptada de https://www.exploit-db.com/exploits/18442/.

ANEXO F. Reporte de vulnerabilidades generado por Nessus.

Nessus vulnerability scanner

# Nessus Scan Report

14/May/2015:20:58:20

## Nessus Home: Commercial use of the report is prohibited

Any time Nessus is used in a commercial environment you MUST maintain an active subscription to the Nessus Feed in order to be compliant with our license agreement.http://www.tenable.com/products/nessus

Table Of Contents



### Hosts Summary (Executive)

[-] Collapse All

[+] Expand All

**172.16.16.135**

**Summary**

| Critical | High | Medium | Low | Info | Total |
|---|---|---|---|---|---|
| 3 | 7 | 14 | 2 | 27 | 53 |

**Details**

| Severity | Plugin Id | Name |
|---|---|---|

| | | |
|---|---|---|
| **Critical (10.0)** | 34477 | MS08-067: Microsoft Windows Server Service Crafted RPC Request Handling Remote Code Execution (958644) (uncredentialed check) |
| **Critical (10.0)** | 35362 | MS09-001: Microsoft Windows SMB Vulnerabilities Remote Code Execution (958687) (uncredentialed check) |
| **Critical (10.0)** | 73182 | Microsoft Windows XP Unsupported Installation Detection |
| **High (9.3)** | 58435 | MS12-020: Vulnerabilities in Remote Desktop Could Allow Remote Code Execution (2671387) (uncredentialed check) |
| **High (7.5)** | 11030 | Apache Chunked Encoding Remote Overflow |
| **High (7.5)** | 11137 | Apache < 1.3.27 Multiple Vulnerabilities (DoS, XSS) |
| **High (7.5)** | 31654 | Apache < 1.3.37 mod_rewrite LDAP Protocol URL Handling Overflow |
| **High (7.5)** | 34460 | Unsupported Web Server Detection |
| **High (7.2)** | 11915 | Apache < 1.3.29 Multiple Modules Local Overflow |
| **High (7.1)** | 11793 | Apache < 1.3.28 Multiple Vulnerabilities (DoS, ID) |
| **Medium (5.1)** | 18405 | Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness |
| **Medium (5.0)** | 12280 | Apache < 1.3.31 / 2.0.49 Socket Connection Blocking Race Condition DoS |
| **Medium (5.0)** | 20007 | SSL Version 2 and 3 Protocol Detection |
| **Medium (5.0)** | 26920 | Microsoft Windows SMB NULL Session Authentication |
| **Medium (5.0)** | 57608 | SMB Signing Required |
| **Medium (4.3)** | 11213 | HTTP TRACE / TRACK Methods Allowed |
| **Medium (4.3)** | 22254 | Web Server Expect Header XSS |

| | | |
|---|---|---|
| **Medium (4.3)** | 26928 | SSL Weak Cipher Suites Supported |
| **Medium (4.3)** | 31408 | Apache < 1.3.41 Multiple Vulnerabilities (DoS, XSS) |
| **Medium (4.3)** | 42873 | SSL Medium Strength Cipher Suites Supported |
| **Medium (4.3)** | 57690 | Terminal Services Encryption Level is Medium or Low |
| **Medium (4.3)** | 57792 | Apache HTTP Server httpOnly Cookie Information Disclosure |
| **Medium (4.3)** | 65821 | SSL RC4 Cipher Suites Supported |
| **Medium (4.3)** | 78479 | SSLv3 Padding Oracle On Downgraded Legacy Encryption Vulnerability (POODLE) |
| **Low (2.6)** | 30218 | Terminal Services Encryption Level is not FIPS-140 Compliant |
| **Low (2.6)** | 31705 | SSL Anonymous Cipher Suites Supported |
| **Info** | 10107 | HTTP Server Type and Version |
| **Info** | 10114 | ICMP Timestamp Request Remote Date Disclosure |
| **Info** | 10150 | Windows NetBIOS / SMB Remote Host Information Disclosure |
| **Info** | 10287 | Traceroute Information |
| **Info** | 10394 | Microsoft Windows SMB Log In Possible |
| **Info** | 10397 | Microsoft Windows SMB LanMan Pipe Server Listing Disclosure |
| **Info** | 10785 | Microsoft Windows SMB NativeLanManager Remote System Information Disclosure |
| **Info** | 10940 | Windows Terminal Services Enabled |
| **Info** | 11011 | Microsoft Windows SMB Service Detection |
| **Info** | 11219 | Nessus SYN scanner |

| | | |
|---|---|---|
| **Info** | 11936 | OS Identification |
| **Info** | 19506 | Nessus Scan Information |
| **Info** | 20094 | VMware Virtual Machine Detection |
| **Info** | 21643 | SSL Cipher Suites Supported |
| **Info** | 22964 | Service Detection |
| **Info** | 24260 | HyperText Transfer Protocol (HTTP) Information |
| **Info** | 24786 | Nessus Windows Scan Not Performed with Admin Privileges |
| **Info** | 25220 | TCP/IP Timestamps Supported |
| **Info** | 26917 | Microsoft Windows SMB Registry : Nessus Cannot Access the Windows Registry |
| **Info** | 35716 | Ethernet Card Manufacturer Detection |
| **Info** | 43111 | HTTP Methods Allowed (per directory) |
| **Info** | 45590 | Common Platform Enumeration (CPE) |
| **Info** | 51891 | SSL Session Resume Supported |
| **Info** | 54615 | Device Type |
| **Info** | 56984 | SSL / TLS Versions Supported |
| **Info** | 66334 | Patch Report |
| **Info** | 70544 | SSL Cipher Block Chaining Cipher Suites Supported |



Fuente: autor del proyecto.

## ANEXO G. Código completo del plugin check_openvas.

```
#!/bin/bash

# Plugin de Nagios para el escaneo de vulnerabilidades de un host
# objetivo con OpenVAS.
#
# Autor: Camilo Alberto Méndez León
# Versión 1.0
# Copyright (C) 2015 Camilo Alberto Méndez León
#
# Los usuarios de este programa poseen los derechos de utilización
# sin restricciones, modificación y redistribución del programa tal
# cual o modificado.

PASSIVE=1
NAME=$(echo $RANDOM $RANDOM | md5sum | cut -c1-8)
TMPFILE=/tmp/$NAME
USER=$1
PASS=$2
TARGET=$3
HOSTNAME=$4
#TYPE=daba56c8-73ec-11df-a475-002264764cea
TYPE=74db13d6-7489-11df-91b9-002264764cea
FORMAT=a3810a62-1f62-11e1-9219-406186ea4fc5

if [ $PASSIVE -eq 1 ]; then
        if [ -z "$HOSTNAME" ]; then
                echo "La sintaxis de comando en modo pasivo es <scan_openvas.sh ompuser passwd IP_TARGET HOSTNAME_NAGIOS>"
                exit 2
        fi
else
        if [ -z "$TARGET" ]; then
```

```
                echo "La sintaxis de comando es <scan_openvas.sh ompuser passwd IP_TARGET>"
                exit 2
        fi
fi

# CREAR TARGET
omp -u $USER -w $PASS --xml='
<create_target>
<name>'$TARGET'</name>
<hosts>'$TARGET'</hosts>
</create_target>' > /dev/null

TARGETID=$(omp -u $USER -w $PASS -T | grep "  $TARGET"$ | awk '{ print $1 }')

# CREAR TASK
TASKID=$(omp -u $USER -w $PASS --create-task --name=$NAME --config=$TYPE --target=$TARGETID)

# START TASK
SCANID=$(omp -u $USER -w $PASS -S $TASKID)

while [ -z "$FINISH" ]; do
        sleep 5
        FINISH=$(omp -u $USER -w $PASS -G | grep $TASKID | grep "  Done  ")
done

# GET REPORT
omp -u $USER -w $PASS --get-report $SCANID --format=$FORMAT > $TMPFILE

# BORRAR TASK
omp -u $USER -w $PASS -D $TASKID

# BORRAR TARGET
omp -u $USER -w $PASS -X '<delete_target target_id="'$TARGETID'"/>' > /dev/null
```

```
# READ REPORT
HIGH=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $2 }')
MEDIUM=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $3 }')
LOW=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $4 }')
LOG=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $5 }')
FALSE_POS=$(grep ^"$TARGET" $TMPFILE | head -n1 | awk '{ print $6 }')

if [ $HIGH -eq 0 ] && [ $MEDIUM -eq 0 ]; then
        OUTPUT=OK
        EXIT=0
fi

if [ $HIGH -eq 0 ] && [ $MEDIUM -gt 0 ]; then
        OUTPUT=WARNING
        EXIT=1
fi

if [ $HIGH -gt 0 ]; then
        OUTPUT=CRITICAL
        EXIT=2
fi

# OUTPUT PLUGIN

if [ $PASSIVE -eq 1 ]; then
        TIMESTAMP=$(date +%s)
        SRV_NAME="Vulnerabilities"
        echo [$TIMESTAMP] PROCESS_SERVICE_CHECK_RESULT";"$HOSTNAME";"$SRV_NAME";"$EXIT";"$OUTPUT: Vulnerabilities: HIGH=$HIGH, MEDIUM=$MEDIUM, LOW=$LOW and LOG=$LOG $(sed -n -e '/Port Summary/,/Security Issues/ p' $TMPFILE | grep -v "Security Issues") "| HIGH=$HIGH MEDIUM=$MEDIUM LOW=$LOW LOG=$LOG" >> /usr/local/nagios/var/rw/nagios.cmd
        exit 0
else
        echo "$OUTPUT: Vulnerabilities: HIGH=$HIGH, MEDIUM=$MEDIUM, LOW=$LOW and LOG=$LOG"
        sed -n -e '/Port Summary/,/Security Issues/ p' $TMPFILE | grep -v "Security Issues"
```

```
        echo "| HIGH=$HIGH MEDIUM=$MEDIUM LOW=$LOW LOG=$LOG"
########        rm $TMPFILE
        exit $EXIT
fi
```

Fuente: autor del proyecto.