UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

TIA
Tecnología, Investigación y Academia

Metodología para el análisis de vulnerabilidades

Vulnerability Assessment Methodology

Germán Alfonso Serrato Polanía[1]

Para citar este artículo: Serrato, G. A. (2016). Metodología para el análisis de vulnerabilidades. *TIA*, 4(2), pp.20-27.

Resumen

El objetivo principal de este trabajo es mostrar una metodología para el rastreo y evaluación de vulnerabilidades en sistemas de gestión de información a nivel lógico, utilizando para este fin herramientas de software gratuito. La metodología propuesta, a manera de guía, consiste en hacer un levantamiento de información para identificar puertos y servicios; acto seguido, se procede a hacer un análisis de vulnerabilidad. Se llega a la conclusión de que el método propuesto tiene un alto sentido práctico y es eficiente para todos aquellos involucrados en el área de seguridad, al utilizar herramientas de fácil acceso con cero costos y que entregan información valiosa y veraz de los componentes de las redes a analizar que proporcionan una mejor visualización y compresión, apoyando la toma de decisiones para mitigar los riegos de seguridad.

Palabras clave: ARP Spoofing, exploits, pentesting, Cross-Site Scripting (XSS).

Abstract

The main aim of this work is to show a methodology for tracking and assessment of vulnerabilities in information systems management at the logical level, using for this purpose free software tools. Proposed methodology, as a guide, consists of gathering information to identify ports and services, to proceed with this information to analyze vulnerabilities. It concludes that proposed method has a high practical sense and it's efficient to all those involved in security area, using easily accessible tools with no charge which provide valuable truthful information on the networks components to be analyzed giving a better display and compression, supporting decision making to mitigate security risks.

Keywords: ARP Spoofing, exploits, pentesting, Cross-Site Scripting (XSS).

Artículo de investigación

Fecha de recepción:
16-05-2016
Fecha de aceptación:
02-11-2016

ISSN: 2344-8288
Vol. 4 No. 2
Julio - Diciembre 2016
Bogotá-Colombia

[1] Estudiante de Especialización en Proyectos Informáticos, Universidad Distrital Francisco José de Caldas. Correo electrónico: serratog@gmail.com

INTRODUCCIÓN

Actualmente casi todos los códigos maliciosos están diseñados para robar algún tipo información; por lo tanto, prácticamente cualquier servicio o recurso protegido por credenciales de acceso podría ser vulnerado si el usuario inicia sesión en un sistema infectado. En estos casos, es importante destacar que el robo de la contraseña ocurre en la computadora de la persona y no en el entorno informático de la empresa proveedora del servicio; podemos tomar como ejemplo el caso de Dorkbot, un gusano que se propagó intensamente en América Latina, que reclutó al menos 80000 computadoras zombis en la región y logró robar más de 1500 cuentas corporativas de correo electrónico. En otras palabras, los cibercriminales responsables de este ataque tuvieron acceso total a la información almacenada en mensajes y archivos adjuntos; esto representa un serio problema para el funcionamiento de la red corporativa, el deterioro de la imagen del negocio y el robo de datos confidenciales [1].

En lo que respecta al sistema de gestión informático se desea blindar sus activos, es decir, los elementos que forman parte del sistema; estos se pueden agrupar en: hardware, que es la parte física del sistema como son servidores, pasarelas, *works station*, sistemas de almacenamiento, cableado (canales de comunicación), etc.; software, que son los sistemas operativos, bases de datos, aplicaciones, protocolos de comunicación, puertos; datos, es decir, la información que es procesada por el software utilizado en el sistema de bases de datos y, en general, todos los paquetes que atraviesan la red; y, finalmente, componentes adicionales, que son los demás actores que interactúan con el sistema de información como son: personas, terceros (proveedores de servicios como servicio de internet, *datacenters*, desarrolladores, *datawarehouse*).

El componente más crítico son los datos, los demás componentes se pueden reemplazar con relativa facilidad; el aseguramiento de los datos va ligado con las políticas de recuperación de desastres asumidos por cada empresa, como copias de seguridad, que preserven la pronta disponibilidad de la información en caso de que se presente algún evento que tenga un impacto negativo, por ejemplo, el fallo de una base de datos.

VULNERABILIDAD: DEFINICIÓN Y CLASIFICACIÓN

Se define vulnerabilidad como una debilidad de cualquier tipo que compromete la seguridad del sistema informático; se pueden agrupar en:

Diseño: a) debilidad en el diseño de protocolos utilizados en las redes, b) políticas de seguridad deficientes o inexistentes; implementación: i) errores de programación, ii) existencia de "puertas traseras" en los sistemas informáticos, c) descuido de los fabricantes; uso: i) mala configuración de los sistemas informáticos, ii) desconocimiento y falta de sensibilización de los usuarios y de los responsables de informática, iii) disponibilidad de herramientas que facilitan los ataques, d) limitación gubernamental de tecnologías de seguridad; vulnerabilidad del día cero: i) se incluyen en este grupo aquellas vulnerabilidades para las cuales no existe una solución conocida, pero se sabe cómo explotarla [2].

Con el fin de protegerse contra este tipo de ataques es importante desarrollar aplicaciones que utilicen lenguajes de programación avanzados, los cuales garanticen una administración precisa de la memoria asignada o que usen un lenguaje de bajo nivel con bibliotecas de función seguras, por ejemplo, las funciones *strncpy*.

Vulnerabilidad de condición de carrera (*race condition*)

Este tipo de vulnerabilidad se presenta cuando múltiples procesos se encuentran en modo

de competición, pues del resultado de la competencia dependerá el orden de ejecución de dichos procesos; al encontrarse los procesos en este tipo de condición no estarán correctamente sincronizados, es aquí donde se manifiesta el riesgo de corrupción de datos. Un ejemplo de esta situación es cuando se presenta el interbloqueo: dos procesos están en espera de que el otro ejecute una acción y, finalmente, ninguno de los dos se ejecuta como consecuencia de esta espera.

Vulnerabilidad de *Cross Site Scripting* (XSS)

Vulnerabilidad que se presenta en aplicaciones web, se caracteriza por la inyección de código VBScript o JavaScript en las experiencias de usuario final al ingresar a alguna página; es una de las vulnerabilidades más conocidas y de mayor de uso siendo el *phishing* su forma más utilizada, la victima cree que está ingresando a una página legítima a través de una URL que normalmente se sugiere en la barra de direcciones; sin embargo, accede a una página diferente a la desea. El riesgo potencial de esta modalidad es que una vez la víctima ingresa sus credenciales son enviadas al atacante.

Vulnerabilidad de denegación del servicio

Su principal objetivo es imposibilitar la prestación de servicios; no busca la captura de claves o datos, busca negar el acceso a servicios y recursos de una organización por un periodo de tiempo no definido. Esto afecta de forma considerable la reputación de las organizaciones, va dirigido a empresas cuyo funcionamiento hace un uso robusto de los recursos de internet para impedir su normal desarrollo; un ejemplo de este tipo de ataques es la saturación de los *call* de comunicación a través de peticiones excesivas.

Este tipo de ataques envían paquetes IP de tamaños y formatos no tradicionales que logran saturar los equipos objetivos generando mal funcionamiento con la consecuente inestabilidad, de esta manera se altera el normal funcionamiento de la prestación de servicios. La recomendación para mantenerse protegido contra este tipo técnicas maliciosas, es visitar regularmente las páginas que informan de nuevos ataques y llevan estadísticas de los mismos; igualmente, es muy importante tener parchados los sistemas operativos que ayudan a mitigar el riego en buena medida.

Vulnerabilidad de ventanas engañosas (Window o ARP Spoofing)

Las ventanas engañosas son las que dicen que se es ganador de tal o cual cosa, lo cual es mentira, estas buscan que el usuario proporcione información. Hay otro tipo de ventanas que si se siguen obtienen datos del ordenador para luego realizar un ataque; su funcionamiento se basa en el envío de falsificados, generalmente *banners*, indicando que se es ganador de un premio, buscan dos objetivos: obtener la información del usuario o datos de identificación del equipo, como MAC, con esto se logra asociar la dirección MAC de un equipo con la dirección de IP de la puerta de enlace o *Gateway*. De esta forma se logra que cualquier información que envíe el usuario no llegue a su destino real sino a la persona que realiza el ataque; así se definen dos tipos de ataque: pasivo, donde se deja pasar la información hacia el destino real y se escuchan los datos enviados; o activo, donde los datos originales son modificados para su posterior reenvío.

Como medida de contención se recomienda el uso de ARP estáticas, la utilización de DHCP Snooping, así se logra detectar si existe una suplantación de ARP.

METODOLOGÍA Y ANÁLISIS

A lo largo de este trabajo se analizará el servidor oficial NTP de Colombia (http://horalegal.sic.gov.co/), utilizando para este fin herramientas gratuitas de software; el procedimiento es a manera de guía y no limita la metodología propuesta a las herramientas que se presentan aquí.

Para la el levantamiento de información se utilizó el servicio *web Robtex*, con esta herramienta se puede obtener bastante información del dominio o IP *address* que se desee analizar; arroja información de redes, rutas, servidores, rutas, información de DNS, sistema autónomo, así como un compendio de listas negras para verificar si la página representa un riesgo.

Scorecard for horalegal.sic.gov.co

Test	Points↓	Date	Extra
Listed in DMOZ	-	Apr 6, 2014	
HostKarma Negatives		May 11, 2014	
HostKarma Positives			
Project Honeypot Negatives			
Project Honeypot Positives			
Spamhaus Negatives			
SORBS Negatives			
MyWOT Negatives			
Google Safe Browsing			
Yandex Safe Browsing			
MyWOT Positives	1/5		Good trustworthiness
Alexa Ranking	2/6	Apr 6, 2014	Alexa rank:68959
Google PageRank	6/10		
Domain Name Service	N/A	Oct 14, 2014 3:31 AM	
Analyzed			
Grand total	9-0=9		

Figura 1. Puntaje de severidad del sitio
Fuente: elaboración propia

En la Figura 1 se puede observar una clasificación según el nivel de riesgo de la página; básicamente se realiza un comparativo contra las bases de datos de los recursos que se observan en la columna de la izquierda, el cual otorga una calificación dependiendo del nivel de riesgo que suponga el dominio analizado; para el caso de estudio, la página es completamente confiable.

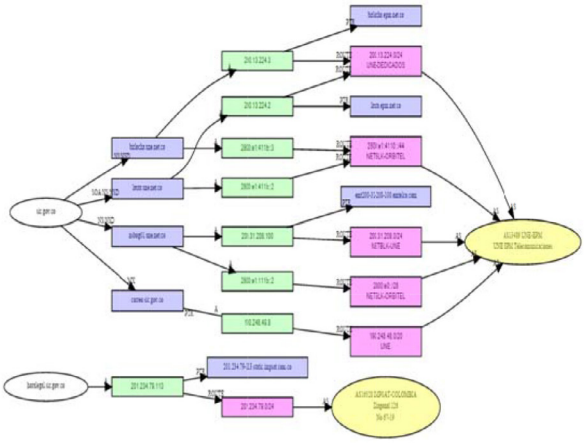

Figura 2. Topología generada por Robtex
Fuente: elaboración propia

Esta herramienta genera un gráfico (Figura 2) que facilita el análisis de cómo están interconectadas las diferentes pasarelas; cada uno de los componentes del mapa tiene hipervínculos, por lo cual el nivel de granularidad es interesante ya que se puede ir más profundo.

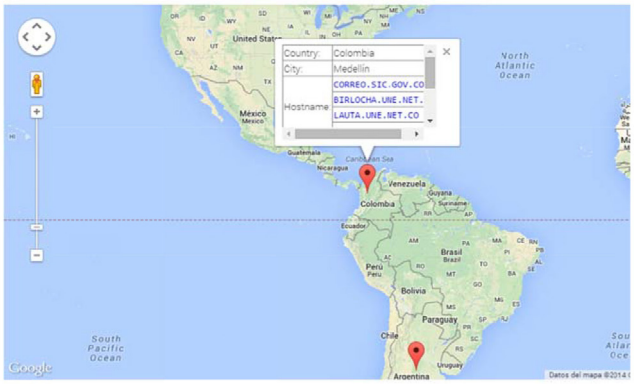

Figura 3. Ubicación geográfica con Robtex. También genera un mapa tipo google que permite ir en profundidad hasta la localización donde se encuentra el origen del dominio. Esta herramienta es muy útil tanto para recolectar información como para saber cómo nos ven desde afuera.
Fuente: elaboración propia

Para el escaneo de puertos y servicio se utilizó *Zenmap*, esta aplicación permite hacer el análisis de los puertos e incluso puede mostrar los servicios que corren a través de dichos puertos; esta

herramienta realiza un *traceroute* y puede traer información del sistema operativo, del dominio objetivo. Es una aplicación de escritorio que se puede instalar tanto en entornos Windows como Linux, Figura 4.

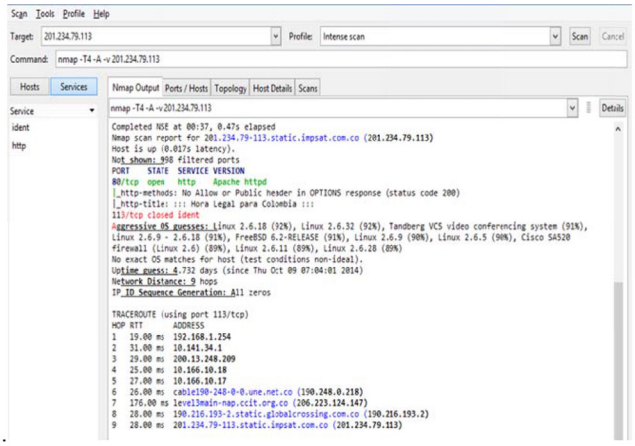

Figura 4. Puertos y servicios con zenmap
Fuente: elaboración propia

En la Figura 5 se puede observar el puerto, el protocolo, el tipo de servicio y la aplicación; también permite observar cuáles puertos están abiertos y cerrados.

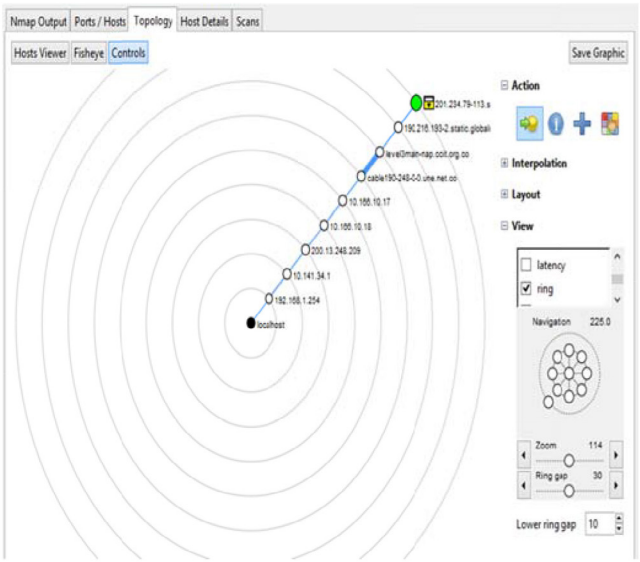

Figura 5. Topología generada con zenmap
Fuente: elaboración propia

Esta herramienta puede generar topologías que permiten un análisis gráfico del *traceroute* y cuenta con la posibilidad de guardar estos mapas (Figura 6).

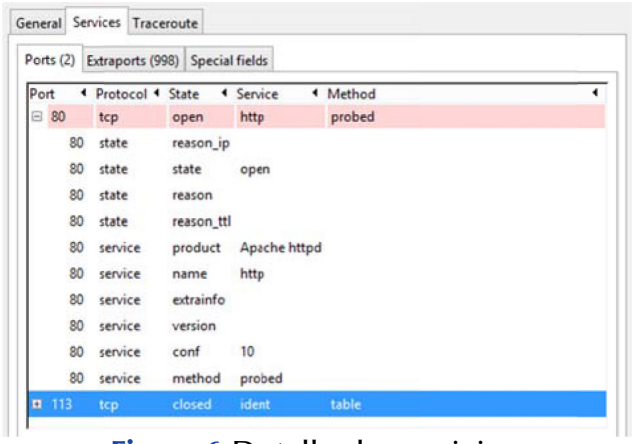

Figura 6 Detalle de servicios
Fuente: elaboración propia

De igual forma, permite ver claramente los puertos utilizados y la información detallada de cada uno de ellos.

Software para análisis de vulnerabilidades

En el presente trabajo se utilizó el programa *nessus* en su versión gratuita, si se desea un software más robusto la siguiente opción es *openvas*, el cual es de distribución libre.

Nessus es un software para análisis de vulnerabilidad, se puede utilizar como *pen tester*, es multiplataforma y ejecuta el escaneo en el sistema objetivo, cuenta con reportes y análisis gráfico de datos; funciona haciendo un escaneo de puertos abiertos para luego ejecutar *exploits* a manera de ataque, los resultados del escaneo pueden ser exportados a distintos formatos como pdf o html. Cuenta con perfiles predeterminados (Figura 7) que facilitan la ejecución de los escaneos; basta con crear una política, generar un escaneo que, dependiendo del nivel de complejidad, tomará más o menos tiempo en ejecutar el análisis.

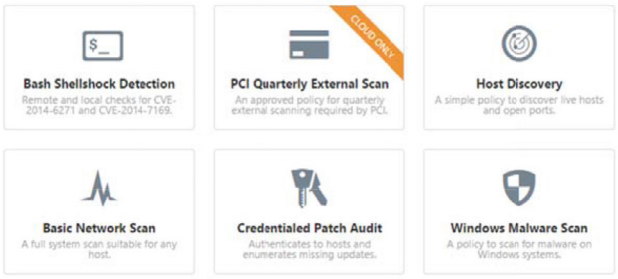

Figura 7. Perfiles de escaneo predefinidos
Fuente: elaboración propia

Muestra el listado de puertos escaneados, los servicios que corren sobre ellos y su nivel de vulnerabilidad o severidad; de igual forma, genera gráficas de tortas para facilitar el análisis de datos, los niveles de severidad se denotan por colores, así pues, el nivel azul muestra un nivel bajo de vulnerabilidad y el color amarillo un nivel medio de vulnerabilidad (Figura 8).

Esta herramienta permite realizar un análisis más profundo, si se da clic sobre cualquiera de los *plug ins* nos mostrará información más detallada (Figura 9), incluso el programa está en capacidad de hacer recomendaciones para mitigar el riesgo de acuerdo a la debilidad que se presente.

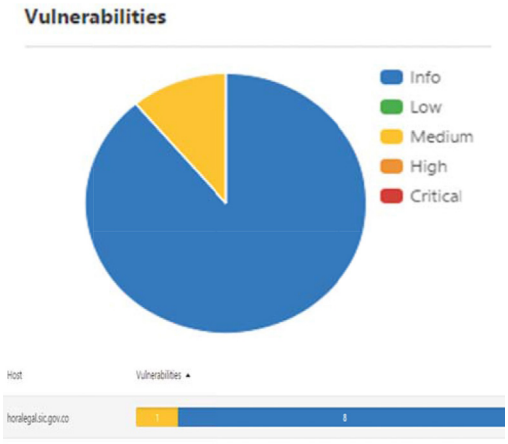

Figura 8. Análisis gráfico de vulnerabilidades con ***nessus***
Fuente: elaboración propia.

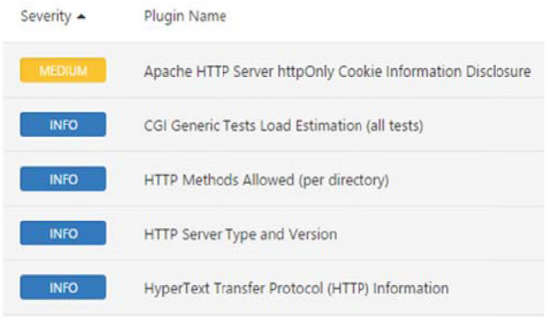

Figura 9. Plugins de escaneo
Fuente: elaboración propia.

A continuación, se puede observar la descripción de la vulnerabilidad que se encontró después de realizado el escaneo.

Description

The version of Apache HTTP Server running on the remote host has an information disclosure vulnerability. Sending a request with HTTP headers long enough to exceed the server limit causes the web server to respond with an HTTP 400. By default, the offending HTTP header and value are displayed on the 400 error page. When used in conjunction with other attacks (e.g., cross-site scripting), this could result in the compromise of httpOnly cookies.

Igualmente entrega la recomendación para mitigar dicha vulnerabilidad.

Solution

Upgrade to Apache version 2.0.65 / 2.2.22 or later (Figura 10).

Figura 10. Nivel de riesgo
Fuente: elaboración propia.

Nessus continene su propio código para generar sobrecarga en los sistemas, como el sqli, el xss y *exploits*, de esta forma simula un ataque y muestra en qué parte encontró la debilidad (Figura 11).

Figura 11. Información de la vulnerabilidad
Fuente: elaboración propia.

CONCLUSIONES

La metodología propuesta tiene un sentido altamente práctico, las herramientas son de fácil acceso y bajo consumo de recursos a nivel de máquina, con excepción de *openvas*; también traen información altamente relevante acerca de cómo se mueve la información en la organización y cómo se ve desde afuera. La información recolectada es confiable y se puede utilizar para todo el tema referente al análisis de vulnerabilidades.

La herramienta de *nessus*, si bien por ser versión gratuita es básica, entrega información muy completa acerca de los puntos débiles que se presentan en la organización; es una herramienta muy útil pues no solo identifica d**ó**nde se encuentran los agujeros en la armadura, sino que también da diagnósticos del porqu**é** la debilidad, además de recomendaciones de cómo solucionarlo, características que siempre se agradecen.

Por supuesto, en el tema de seguridad ayudan mucho los buenos hábitos, así que si se está a cargo de esta, y aún si no lo est**á,** es bueno consultar periódicamente p**á**ginas como google hacking donde se podrán encontrar estadísticas acerca de técnicas de ataques; de igual forma, cuenta con herramientas que pueden ayudar a fortalecer los sistemas de gestión de información como son el análisis de metadatos.

REFERENCIAS

[1] Goujon, A. (2013). ¿El fin de las contraseñas? ESET Latinoamérica Recuperado de: http://www.welivesecurity.com/wp-content/uploads/2014/01/doble_autenticacion-_el_fin_de_las_contrasenas.pdf

[2] Mifsud, E. (s.f.). *Introducción a la seguridad informática-Vulnerabilidades de un sistema informático*. Gobierno de España: Observatorio tecnológico. Recuperado de: http://recursostic.educacion.es/observatorio/web/gl/software/software-general/1040-introduccion-a-la-seguridad-informatica?start=3

[3] Moreno, A. (s.f.). *Amperisblog*. Recuperado de: http://amperis.blogspot.com/2008/09/robtex-com.html

[4] Gordon, L. (s.f.). *Nmap.org*. Recuperado de: http://nmap.org/zenmap/

[5] Openvas.org. (s.f.). *Project News*. Recuperado de: http://www.openvas.org/news.html#openvas7

[6] Dragon, N.(2011, 17 de octubre). *FOCA–Herramienta para análisis de Meta Datos*. Recuperado de: http://www.dragonjar.org/foca-herramienta-para-analisis-meta-datos.xhtml

[7] López, E. (2011). *Google Hacking para Pentester*. Mexico: s.n.

[8] Foundation, OWASP (s.f.). *OWASP Zed Attack Proxy Project*. Recuperado de: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

[9] Franco, D., Perea, J. y Tovar, L. (2013). Herramienta para la detección de vulnerabilidades

basada en la identificación de servicios. *Inf. tecnol. 24*(5), doi: http://dx.doi.org/10.4067/S0718-07642013000500003

[10] Patiño, D. (s.f.). *Buena información de redes*. Recuperado de: http://diegopulgarinsena.blogspot.com/2012/04/instalacion-de-nessus-en-windows.html

[11] González, J. (2013). ¿Qué es Google Hacking? Recuperado de: http://www.seguridadparatodos.es/2013/07/google-hacking-parte-i.html

[12] Pelaez, J. (s.f.). *Inteco*. Recuperado de: http://www.inteco.es/blogs/post/Seguridad/BlogSeguridad/Articulo_y_comentarios/empresa_vulnerable_ataque_informatico

[13] Agrawal, A. (s.f.). *eXploit prot0col blog*. Recuperado de: http://exploitprotocol.blogspot.com/2013/10/pentester-academy-web-application_30.htm