Vulnerabilidades en HTTP/2

Mauro F. Fornaroli [(1)], Mónica D. Tugnarelli [(1)], Sonia R. Santana [(1)], Javier Díaz [(2)]
[(1)] Facultad de Ciencias de la Administración – Universidad Nacional de Entre Ríos
[(2)] Facultad de Informática – Universidad Nacional de La Plata
e-mail: maufor, montug [@fcad.uner.edu.ar]

Resumen

El PID 7052 tiene por objetivo obtener conclusiones generales y comparativas acerca de dos metodologías de recolección de datos digitales, una a priori y otra a posteriori de un evento de seguridad analizando particularmente su performance en entornos de servidores web. Como parte de las actividades previstas en este proyecto se realiza un análisis exploratorio del protocolo HTTP, comparándose las versiones HTTP 1.0 y 1.1 y la nueva HTTP/2 e identificando puntos de control sobre los cuales se realiza la captura de tráfico y recolección de datos. Se consideran aspectos de seguridad, concentrándose en el estudio de vulnerabilidades conocidas de HTTP/2. Además, se define y configura un entorno de testing, utilizando herramientas Open Source, para simular un ataque de Denegación de Servicio Distribuido (DDoS) a los fines de observar el comportamiento del servidor y el volumen de datos resguardados en logs y para analizar la performance de las metodologías mencionadas en instancias de recuperación del servicio y resguardo de la evidencia digital.

Palabras clave: HTTP, seguridad, vulnerabilidades, DDoS

Contexto

El Proyecto de Investigación y Desarrollo PID-UNER 7052 para Director Novel con Asesor, denominado "Análisis de Metodologías de Recolección de datos digitales" se encuadra en una de las líneas de investigación establecidas como prioritarias para su fomento, "Arquitectura, Sistemas Operativos y Redes", de la carrera Licenciatura en Sistemas de la Facultad de Ciencias de la Administración. Se adecua, además, a las prioridades de la Universidad Nacional de Entre Ríos por ser un proyecto aplicado a la investigación sobre Tecnologías de la Información y la Comunicación [1].

Introducción

HyperText Transfer Protocol (HTTP) es un protocolo de nivel de aplicación con características definidas para utilizarse en sistemas de información distribuidos, colaborativos e hipermediales. Se han introducido mejoras en sus diferentes versiones sobre todo tendientes a optimizar la perfomance del mismo, reducir el consumo de recursos y la latencia y para resolver algunos inconvenientes que tiene la comunicación a través TCP [2][3][4][5].

La última versión HTTP/2 [6], presenta un protocolo binario que incorpora

multiplexación y el uso obligatorio de TLS (Transport Layer Security) conservando la misma semántica y la compatibilidad con las versiones 1.0 y 1.1 El protocolo se implementa si el cliente y el servidor tienen soporte y en el caso de que alguno de los dos no lo tengan, en la negociación de protocolo, se acuerda usar las versiones anteriores. Actualmente la mayoría de los browsers y entornos de servidor cuentan con implementaciones oficiales para esta nueva versión.

Entre las características más relevantes del protocolo HTTP/2 pueden mencionarse [6][7][8]:

- Formato de mensajes basados en tramas;
- Reducción de la latencia al permitir una multiplexación completa de streams de solicitudes y respuestas;
- Disminución de la sobrecarga de protocolo mediante la compresión eficiente y codificación binaria HPACK (Header Compression for HTTP/2);
- Incorporación de soporte para la prioridad de solicitudes y capacidades "Server Push";
- Definición de nuevos mecanismos de control de flujo, manejo de errores y actualizaciones.

Dado que HTTP es considerado un protocolo no seguro, en el desarrollo de HTTP/2 se identificaron y gestionaron los riesgos de seguridad del nuevo protocolo, tanto en base a decisiones de diseño como a guías de implementación. Sin embargo, algunas implementaciones de servidores HTTP/2 pueden no seguir estrictamente estas guías, tornándose vulnerables a distintos tipos de ataques de Denegación de Servicio (DoS) [8].

Algunas vulnerabilidades conocidas son:

- Stream Reuse,
- Slow Read Attack,
- Dependency Cycle DoS,
- HPACK Bomb.

Resulta de interés entender como este tipo de ataques puede afectar el comportamiento del servidor y para analizar la perfomance de las metodologías de recolección de datos digitales en instancias de recuperación del servicio y resguardo de la evidencia digital.

Líneas de Investigación, Desarrollo e Innovación

Con este proyecto de investigación se espera conformar una base de conocimiento sobre la forensia informática en relación a metodologías de recolección de datos digitales. Como parte de las actividades se desarrolla un estudio exploratorio de las principales características de HTTP como aporte para la identificación de puntos de control de este protocolo y su comportamiento ante incidentes de seguridad.

Resultados y Objetivos

El objetivo principal del PID 7052 es obtener conclusiones generales y comparativas acerca de las metodologías de recolección de datos a priori de un evento de seguridad y de recolección de datos a posteriori de un evento de seguridad,

analizando particularmente su performance en entornos de servidores web.

Con el desarrollo de las actividades previstas en el proyecto, se obtuvieron los siguientes resultados:

Identificación y descripción de puntos de control en protocolos HTTP. Se analizaron y compararon las versiones del protocolo HTTP 1.0 y 1.1 y la nueva HTTP/2, confeccionándose una tabla [9] que destaca sus principales prestaciones y diferencias entre ellas (Tabla 1). Luego, se identificaron los puntos de control para el protocolo HTTP, que en este proyecto se implementó en un servidor web Apache, sobre los cuales se realizó la captura de tráfico y recolección de datos. Se controlaron los siguientes puertos y archivos:

- puertos 80 y 443: que registra el tráfico entrante y saliente del protocolo HTTP y asociados como SSL - *Secure Sockets Layer*.
- archivos log del sistema operativo Ubuntu Linux (/var/log/): messages.log, auth.log, secure, tmp/wtmp
- Archivos log de Apache: error.log y access.log.
- archivos de configuración del servidor Apache

Versión	HTTP/1.0 (1996)	HTTP/1.1 (2000)	HTTP/2 (2015)
RFC	*RFC 1945*	*RFC 2616*	*RFC 7540*
Manejo de requerimientos	Un requerimiento entregado por vez sobre una conexión.	Mecanismo HTPP Keep Alive: varios requerimientos pueden utilizar múltiples conexiones con el servidor para reducir la latencia.	Múltiples mensajes de requerimiento/ respuesta sobre una misma conexión. Permite asignar prioridades a los requerimientos.
Header	Formato texto	Formato texto	Formato binario. . Compresión de header (Algoritmo HPACK).
Multiplexación	No permite conexiones simultáneas	No permite conexiones simultáneas	Permite múltiples solicitudes y respuestas en paralelo usando la misma conexión TCP, enviando cada requerimiento en un stream diferente.
Servidor	Descarga de recursos a solicitud del cliente (primero HTML, luego CSS, JS, imágenes, enlaces)	Descarga de recursos a solicitud del cliente (primero HTML, luego CSS, JS, imágenes, enlaces)	Tecnología server push: Permite cargar los archivos (CSS, JS, imágenes) desde el servidor al cliente sin que éste lo pida.

Tabla 1. Principales diferencias entre versiones HTTP

Definición y Configuración de entornos de testing. Se configuró un entorno de trabajo en una red LAN Ethernet conformada por un servidor, con Sistema Operativo Ubuntu Server [10] versión 15.10 y un Servidor Web Apache Server [11] versión 2.4.12. Se dispuso de una notebook dedicada con el toolkit Kali Linux versión 64bit 2017.1 [12]. Como marco de trabajo y guía general de buenas prácticas para las pruebas de laboratorio se seleccionaron las especificaciones RFC 3227 [13], ISO/IEC 27037:2012[14], OSSTMM (Open Source Security Testing Methodology Manual) [15]

Simulación de ataque de Denegación de Servicios (DoS). Empleando herramientas de hacking se realizó la simulación de un ataque DDoS, con metodología de inundación, a los fines de observar el

comportamiento del servidor, el volumen de datos resguardados en logs y para analizar la perfomance de las metodologías de recolección de datos digitales en instancias de recuperación del servicio y resguardo de la evidencia. Para darle la característica de distribuido, se emplearon varias máquinas virtuales actuando en paralelo. Los ataques se realizaron tanto desde el interior como del exterior de la red local.

Como avances y a modo de conclusiones parciales de estas etapas, se puede establecer que:

1. HTTP/2 permite establecer puntos de control para analizar su comportamiento ante la recolección de datos frente a un incidente de seguridad.

2. Las herramientas de forensia informática open source, como las usadas en este PID, brindan el soporte necesario y eficiente tanto para implementar un seguimiento preventivo que incluya recolección de datos como para un análisis forense luego de un incidente de seguridad.

3. Los puntos de control del protocolo HTTP, representados en su mayor parte por archivos log, muestran un crecimiento exponencial al momento en que se realizó la simulación del ataque DDoS, por lo que requiere pensar en una infraestructura adecuada.

4. Las vulnerabilidades presentes en HTTP/2 facilitan nuevas variantes de DoS. Por ejemplo, mientras que el atacante en HTTP 1.x tenía que abrir tantas conexiones TCP como el servidor de la víctima, en HTTP / 2 los ataques se vuelven más simples, ya que se puede utilizar la multiplexación de flujos de datos.

Formación de Recursos Humanos

Este proyecto prevé la formación e iniciación en actividades de investigación de tres docentes de la carrera Licenciatura en Sistemas, la formación de dirección en proyectos de un docente y la realización de una tesis de maestría correspondiente a la Maestría en Redes de Datos de la Facultad de Informática de la UNLP.

Referencias

[1]. Tugnarelli, M., Fornaroli, M., Santana, S., Jacobo, E., Díaz, J.: Análisis de Metodologías de Recolección de Datos Digitales. Libro de Actas Workshop de Investigadores en Ciencias de la Computación 2017, pp. 1000-1004. ISBN 978-987-42-5143-5.

[2]. RFC 1945 Hypertext Transfer Protocol-HTTP/1.0 http://tools.ietf.org/html/rfc1945

[3]. RFC 2616 Hypertext Transfer Protocol-HTTP/1.1 http://tools.ietf.org/html/rfc2616

[4]. James F Kurose, Keith W. Ross. Redes de Computadoras. Un Enfoque descendente basado en Internet, 2da. Edición, Pearson Educación, 2004

[5]. Balachander Krishnamurthy, Jeffrey C. Mogul y David M. Kristol. Key Differences between HTTP/1.0 and HTTP/1.1.

[6]. RFC 7540 Hypertext Transfer Protocol Version 2 (HTTP/2) https://tools.ietf.org/html/rfc7540

[7]. Daniel Stenberg. HTTP2 Explained. ACM SIGCOMM Computer Communication Review, Volume 44, Number 3, July 2014.

[8]. Imperva. Hacker Intelligence Initiative. HTTP/2:In-depth analysis of the top four flaws of the next generation web protocol. 2016

[9]. Mónica D. Tugnarelli, Mauro F. Fornaroli, Sonia R. Santana, Eduardo Jacobo, Javier Díaz: Análisis de metodologías de recolección de datos digitales en servidores web. Libro de Actas. XXIII Congreso Argentino de Ciencias de la Computación CACIC 2017. VI Workshop de Seguridad Informática, pp. 1230-1238. ISBN 978-950-34-1539-9.

[10]. Ubuntu Server https://www.ubuntu.com/server

[11]. Apache Server https://httpd.apache.org/

[12]. *KALI Linux. www.kali.org*

[13]. RFC 3227 Guidelines for Evidence Collection and Archiving. https://www.ietf.org/rfc/rfc3227.txt

[14]. Guidelines for identification, collection, acquisition and preservation of digital evidence" ISO/IEC 27037:2012

[15]. Open Source Security Testing Methodology Manual (OSSTMM) http://www.isecom.org/mirror/OSSTMM.3.pdf