Universidad Autónoma Metropolitana
Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Diseño e implementación de un servidor de balance de cargas con bajos requerimientos en hardware para la red del Consejo de Salubridad General

Modalidad: Estancia Profesional

Trimestre 2017 Otoño

Alumno:

Gerardo González Aguilar
Matrícula: 2123029383

Asesores del Proyecto:

M. en C. José Alfredo Estrada Soto
Mtro. Juan Manuel Saucedo Camacho

11 de diciembre de 2017

1 Declaratoria

Yo, **José Alfredo Estrada Soto**, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Mtro. José Alfredo Estrada Soto

Yo, **Juan Manuel Saucedo Camacho**, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Mtro. Juan Manuel Saucedo Camacho

Yo, **Gerardo González Aguilar**, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Gerardo González Aguilar

Resumen

El presente reporte de Estancia Profesional contiene el informe de las actividades realizadas para el Consejo de Salubridad General (CSG) por medio del Departamento de Sistemas del mismo.

En la sección de desarrollo se describe cómo se realizó este proyecto. Primeramente creando un análisis de su red de datos, el cual permitió establecer las políticas necesarias para el balance de carga. Estas políticas sirvieron para crear los archivos de inicio que contienen la configuración precisa para detectar y prevenir posibles ataques a los servidores web. Todo esto para mejorar y proteger el acceso a la información del CSG contra saturaciones y el uso mal intencionado.

Tabla de Contenido

Índice de Figuras

1. Introducción

El acceso eficaz a la información es una característica trascendental en el ámbito de las redes de datos. Los usuarios requieren que el acceso a esta información tenga alta disponibilidad, fiabilidad y eficiencia. Por ello, las instituciones tanto públicas como privadas necesitan soluciones de *hardware* y de *software* que les permitan alcanzar un buen rendimiento en sus redes de datos. Una alta demanda de información sin la debida administración puede ocasionar fallos en las redes de datos e incluso riesgos de seguridad.

Cuando existe una carga importante en un servidor, es posible que su rendimiento disminuya, pues los recursos disponibles deben ser divididos entre más peticiones. Para aliviar esto, muchas instituciones utilizan más de un servidor de tipo espejo para alojar sus datos. La administración de peticiones a estos servidores la realiza un Servidor de Balance de Carga[1] (*Load Balancer Server*). El servidor de balance distribuye las peticiones de manera transparente y alivia la congestión en las líneas de datos. Esto reduce retrasos y fallas.

Otro beneficio importante de los servidores de balance es protección contra ataques de negación de servicio y abusos de usuarios. Las políticas de balance previenen que se haga un uso indebido de los recursos del servidor. Ya que es necesario tener una alta disponibilidad de la información, el servidor de balance identifica correctamente a un usuario ordinario de un abusador. Esto mejora la experiencia para el resto de los usuarios y previene que se bloquee el acceso a los datos. En esta estancia profesional se participa en un proyecto de un servidor de balance de carga el cual busca mejorar el desempeño del servidor web en la red del Consejo de Salubridad General (CSG). A través de un análisis y políticas personalizadas, se mejora el desempeño y seguridad de la red que ayudará a aumentar la calidad de servicio y por ende, el acceso a la información.

[1] Los balanceadores de carga pueden ser soluciones de *hardware*, *software* o ambas que distribuyen el tráfico de red entre dos o más equipos que proveen algún servicio (servidores).

2. Antecedentes

Artículos

- *Dynamic Load Balancing in a Distributed Computer System* [1]

En los últimos años, el incremento de las redes computacionales y servicios basados en la nube también ha incrementado el tráfico de red. La idea de un servidor de balance de cargas centralizado no es nueva, de hecho, surgió casi a la par de las redes de datos modernas. Una de las referencias más tempranas es en este discurso de balanceo en 1991. Aquí se menciona que un dispositivo central mantiene una tabla de cargas en la cual se decide cómo transferir las cargas entre nodos de alta y baja demanda. Este es el concepto básico que siguen todos los balanceadores de carga actuales, sin importar el algoritmo de balance o el tamaño de la red. Debido a que en la red actual del CSG no existe un servidor de balance de carga, existen varios problemas de acceso de datos (por ejemplo, retrasos o congestión de la red). Estos son los mismos problemas que se menciona existían antes del concepto de balance de carga, y que posteriormente fueron solucionados gracias a este servicio.

- *Load Balancer as a Service in Cloud Computing* [2]

Con el paso del tiempo, los Centros de Datos se volvieron más comunes, lo que propició que el cómputo en la nube se consolidara como parte esencial de las redes de datos. Los servidores de balance de carga fueron, a su vez, mejorados pero en pocas ocasiones son vistos como un servicio esencial de la red. En este artículo, se aborda el tema de balanceadores de carga como un servicio y no como un producto de hardware. Este artículo tiene algunos puntos en común con el proyecto a realizar. Primeramente, un servidor de balance de carga como un servicio de red y no como un producto de hardware, es similar al enfoque de un servidor que opere con bajos recursos. De esta manera, existe un énfasis en el servicio de balance y no en las características físicas del hardware de procesamiento. Esto puede verse reflejado en el uso de herramientas que optimicen los recursos o tengan bajos requerimientos en hardware. Lo que nos lleva al segundo punto en común. Al catalogarse al balance de carga como servicio, se le considera un elemento indispensable en la misma categoría que la telefonía o el servicio de internet, los cuales son considerados parte esencial para las actividades del CSG. Un balance de carga proveerá un servicio imprescindible para el oportuno acceso a los datos del consejo, más específicamente para su página web.

- *Load Balancing in Cloud Data Center using Modified Active Monitoring Load Balancer* [3]

En este artículo del 2016, se expone cómo un algoritmo personalizado de balance de red se compara con otras soluciones actualmente disponibles. La principal diferencia con el proyecto es que el balance lo realizan entre máquinas virtuales y no entre servidores físicos; sin embargo, es una aproximación similar a lo que se desarrolla en este proyecto. El uso de una configuración (un algoritmo en este caso) personalizada resultó ser la mejor opción.

Aunque ya se ha decidido utilizar el algoritmo de "Leastconn" en el servidor de balance de carga (el algoritmo que selecciona el nodo con menos conexiones), se realizará una comparación y prueba de los algoritmos disponibles en el software HAProxy [4] de

balance de carga. Esto es para mostrar las diferencias, o falta de ellas, en cuanto a tiempos de acceso y rendimiento de los distintos algoritmos disponibles con datos reales. Otras configuraciones personalizadas también serán implementadas.

- *DDoS attacks and defense mechanisms: classification and state-of-the-art* [5]

Los ataques de negación de servicio se han vuelto una de las mayores amenazas para las redes de datos actuales. Este documento habla de cómo pueden realizarse con relativamente pocos recursos. Sin embargo, también indica algunas de las mejores prácticas para prevenir o mitigar estos ataques. Esto se relaciona directamente con uno de los puntos clave de la propuesta actual: la seguridad. Este texto menciona dos prácticas importantes para mejorar la seguridad. En primer lugar, deshabilitar servicios no utilizados ayudará a prevenir ataques a protocolos específicos. En segundo lugar, el uso de un servidor de balance ayudará a mitigar ataques o, de ser necesario, cambiar el servidor activo en caso que el principal sea deshabilitado por el ataque.

Dependiendo de la configuración del servidor de balance, se pueden analizar patrones de ataque para prevenir nuevos ataques. De esta manera, se reforzará aún más la seguridad de la red.

Tesis

- Algoritmos de Balance de Carga con Manejo de Información Parcial [6]

Como ya se mencionó, los algoritmos y configuraciones personalizadas tienen una gran ventaja sobre las opciones disponibles por defecto en los *software* de balance. En esta tesis, se proponen dos algoritmos personalizados de balance de carga con manejo de información parcial. Los algoritmos personalizados tienen mucho mejor desempeño que los tradicionales. Sin embargo, el balance que utilizan es distribuido a diferencia del balance centralizado que se plantea en esta propuesta. Al ser un balance centralizado se reducen los puntos de falla y no se depende de la información de nodos adyacentes.

- Análisis y Gestión de Recursos para Brindar Seguridad en una Red Empresarial [7]

El reforzamiento de la seguridad de una red requiere un análisis profundo de los recursos, infraestructura y dispositivos disponibles. Esta tesis se enfocó a evaluar y mejorar el rendimiento y seguridad de una red empresarial mediante la implementación de políticas de seguridad y *hardware* de nueva generación. Esto es comparable a la actual propuesta, la cual implementará políticas de balance que ayudarán al rendimiento y seguridad de la red. También la implementación de un dispositivo físico (*hardware)* que gestionará todas estas políticas de balance.

3. Justificación

El CSG siendo un órgano normativo consultivo, requiere de una infraestructura adecuada a las necesidades del Plan Nacional de Desarrollo [8]. Es por ello que el acceso a la información y servicios ofrecidos por el Consejo debe ser fiable, flexible y transparente.

Un servidor de balance de cargas, con bajos requerimientos en hardware, apoyaría a la infraestructura de red para prevenir la saturación de los canales de información. Dicho servidor es inexistente en la configuración actual de la red. Lo anterior se genera por los lineamientos actuales aplicables del gobierno federal, evitándoles adquirir equipo especializado por medio de una partida presupuestal.

Los dispositivos de balance de carga disponibles comercialmente no ofrecen el nivel de personalización y modularidad que brindará este servidor. Esto es primordial para reforzar la configuración de seguridad en la red del CSG, la cual no previene la saturación de la red ni dificulta los ataques de negación de servicio.

4. Objetivos

Objetivo General

Diseñar e implementar un servidor de balance de cargas que mejore el rendimiento y seguridad de la red del Consejo de Salubridad General

Objetivos Específicos

- Diseñar y construir el Módulo de Generación de Estadísticas
- Diseñar y construir el Módulo de Políticas de Balance del Servidor
- Diseñar el Módulo de Acreditación de Software

5. Marco Teórico

Servidores

Los servidores son poderosas computadoras que proveen información o un servicio a una red de computadoras llamadas clientes. Por lo general, los servidores Web son los anfitriones de las páginas web en internet [9].

Red de computadoras

Es un gran número de computadoras separadas pero interconectadas entre sí para compartir recursos y datos [9].

Modelo OSI (*Open Systems Interconnection*)

El modelo OSI está basado en una propuesta desarrollada por la Organización Internacional de Normas (ISO) en 1995 para la estandarización de protocolos usados en las diversas capas de un sistema de computadoras [9]. El modelo OSI tiene siete capas que se resumen en la siguiente Figura 1.

Figura 1. Capas del Modelo OSI con los protocolos asociados a ellas

La capa de aplicación, transporte y de red es donde se ubican los protocolos de interés para este proyecto. La capa de transporte divide los datos de las capas superiores en unidades más pequeñas llamados paquetes, mientras que la capa de red se encarga de encaminar los

paquetes por medio de rutas. Es sobre la capa de red donde trabajará el balance de carga redirigiendo las peticiones.

Protocolo HTTP

El Protocolo de Transferencia de HiperTexto (*HyperText Transfer Protocol*) es un protocolo simple de solicitud-respuesta que por lo general opera sobre TCP. Especifica qué mensajes pueden enviar los clientes a los servidores, y qué respuestas reciben de estos mensajes.

Los encabezados de solicitud y respuesta se proporcionan en ASCII[2], justo igual que en protocolo SMTP. Este modelo simple fue en parte responsable del éxito anticipado de la web, ya que simplificó los procesos de desarrollo e implementación [9].

Protocolo TCP y UDP

El Protocolo TCP (Protocolo de Control de la Transmisión, del inglés *Transmisión Control Protocol*), es un protocolo confiable orientado a la conexión que permite que un flujo de bytes originado en una máquina se entregue sin errores a cualquier otra máquina en la interred.

UDP (Protocolo de Datagrama de Usuario, del inglés *User Datagram Protocol*), es un protocolo sin conexión, no confiable para aplicaciones que no desean la asignación de secuencia o el control de flujo de TCP y prefieren proveerlos por su cuenta [9].

Seguridad en Redes

Se refiere a la protección de los datos que fluyen a través de una red en sus diferentes capas o niveles. Esto no solo se refiere a los accesos no autorizados sino también a la integridad de los datos transmitidos. Cada capa posee mecanismos de protección de datos. Por ejemplo, en la capa de datos los paquetes pueden encriptarse en el origen y desencriptarse cuando lleguen a su destino. En la capa de red se pueden instalar contrafuegos que son dispositivos o sistemas que controlan o impiden el tráfico de paquetes nocivos. En la capa de transporte se pueden encriptar conexiones enteras que pueden garantizar la máxima seguridad [9].

Para garantizar un alto nivel de seguridad, varias de estas tecnologías de protección deben usarse en conjunto y de manera apropiada. Una configuración mal realizada puede ocasionar más riesgos de seguridad de los que se busca prevenir.

Balance de Carga (*Network Load Balancing*)

El balance de carga o equilibrio de red es una característica que distribuye el tráfico del protocolo TCP/IP por distintos servidores. Estos servidores se combinan en un solo clúster virtual de dos o más equipos que ejecutan aplicaciones o servicios, normalmente del tipo *web*. A los servidores de un clúster se denominan *hosts*, y cada uno ejecuta una copia

[2] El ASCII es, básicamente, un código de caracteres que tiene su base en el alfabeto romano o latino.

del servidor principal. Es posible configurar la carga que cada servidor puede manejar, el tipo de balance a utilizar entre otros parámetros. Se pueden agregar más *hosts* dinámicamente para una mejor administración de la carga.

El balance de carga permite que todos los dispositivos del clúster puedan ser accesados por un solo conjunto de direcciones IP. Cuando se produce un fallo en un equipo, la carga se redistribuye de manera automática entre los equipos restantes. Después cuando el equipo sea restablecido, este se puede unir nuevamente al clúster y tener la misma carga asignada anteriormente. El servicio de balance garantiza que las aplicaciones y servicios permanezcan disponibles por mucho más tiempo, el tiempo de inactividad se reduce al mínimo y que los sistemas sean más escalables [10].

Alta disponibilidad

Es cuando un sistema proporciona un servicio aceptable y con poco tiempo de inactividad. La alta disponibilidad en el servicio de balance de carga incluye características para detectar de manera automática:

- Cuando un host se desconecta del clúster por alguna falla
- Equilibrar la carga cuando se agregan más hosts
- Redistribuir la carga en un periodo corto de tiempo

Escalabilidad

La escalabilidad permite que un servicio pueda aumentar su capacidad y cubrir una demanda mayor. En este caso del balance de carga, es posible agregar más servidores o *hosts* de manera gradual para aumentar el tamaño del clúster. De igual manera es posible reducir el número de *hosts* cuando la carga disminuya.

Administración

La capacidad de administrar los clústeres se realiza desde un punto central, donde se puede especificar el comportamiento del balance de carga, el algoritmo de balance, el control de acceso para ciertas IPs entre otros.

Seguridad

Una característica del balance de carga es que permite proteger los hosts ante abusos o ataques a sus líneas de datos. Un alto número de solicitudes a un determinado clúster puede ocasionar un bloqueo o saturación de sus líneas. El balance de carga permite filtrar, bloquear y redirigir peticiones hechas dependiendo de sus características.

Ataques de Negación de Servicio Distribuidos (*Distributed Denial of Service Attack - DDoS*)

Los ataques de negación de servicio distribuidos son similares a los ataques de negación de servicio (DoS), sin embargo, la principal diferencia es que el cliente de DDoS

corre desde varios dispositivos. En estos ataques, los dispositivos mandan una gran cantidad de paquetes TCP/IP a un destino conocido, lo que impide a usuarios legítimos acceder a los recursos o el servidor motivo del ataque [11].

Por lo general, los dispositivos que realizan el ataque no saben que han sido infectados por un *malware* o virus. El *software* malicioso contiene el código necesario para realizar estos ataques. A este grupo de dispositivos infectados se les denomina *botnet*.

Redes Zombi (*Botnet*)

Las redes zombi son aquellas que se conforman por equipos infectados por un virus troyano. Por lo general, el virus viene dentro de programas que ofrecen un producto o servicio (por ejemplo números de lotería) de manera gratuita. Una vez ejecutado, el programa malicioso instala un cliente que utiliza los recursos del dispositivo para realizar fraudes, ataques de negación de servicio, reclutar más dispositivos o todos los anteriores.

En los últimos años, se ha dado un incremento de redes *botnet* conformadas por dispositivos IoT[3]. A diferencia de *botnet* tradicionales, las redes constituidas de dispositivos IoT utilizan vulnerabilidades en el *firmware* de los dispositivos para instalar código malicioso. En algunos casos es posible prevenir la infección al cambiar los valores de fábrica.

En octubre de 2016, hubo un ataque DDoS a la red de servidores Dyn. Estos servidores proveen servicio DNS[4] a otras compañías como Twitter y Netflix. Aproximadamente 100,000 dispositivos IoT como cámaras IP participaron en el ataque y se espera que este número siga aumentando mientras más dispositivos se conecten al internet [12].

Otro uso común de las redes zombi es para actividades fraudulentas de publicidad; por ejemplo, si un sitio muestra anuncios monetizados, estas redes son usadas para crear "*clicks*" fantasma y así generar visitas falsas a dichos sitios.

Linux CentOS

Los sistemas operativos para servidores por excelencia son las diferentes distribuciones de Linux, más específicamente Linux CentOS. Debido a su flexibilidad, seguridad y escalabilidad CentOS provee un ambiente amigable para administradores de redes y sistemas. Linux CentOS está basado en *Red Hat Enterprise Linux* (RHEL) y se mantiene como una plataforma gratuita distribuible. Una de las principales ventajas de CentOS es que tiene soporte para sus distribuciones hasta por 10 años; la distribución actual, CentOS 7, tendrá actualizaciones de mantenimiento hasta el año 2024 [13].

[3] IoT (*Internet of Things*) se refiera la interconexión de dispositivos cotidianos al internet. Esto incluye cámaras, sensores y electrodomésticos.
[4] DNS (*Domain Name System*) es un servicio de resolución de nombres que traduce nombres de domino a IPs.

De acuerdo al sito top500.org, CentOS se ubica como el segundo sistema operativo más utilizado en servidores y supercomputadoras. El primer lugar se comparte por otras distribuciones de Linux [14].

HAProxy (*High Availability Proxy*)

Dentro de los software de balance de carga, HAProxy es el software de código abierto número uno a través de diversas plataformas [4]. HAProxy es usado por varios sitios de alto perfil como Github, Reddit, StackOverflow, Tumblr entre otros. Las principales características del *software* son la alta disponibilidad, escalabilidad, flexibilidad y seguridad. Estas características nos aseguran un óptimo rendimiento de la red y los servicios que se administran.

HAProxy posee algoritmos avanzados que permiten acelerar trafico HTTP mediante compresión y enrutamiento. Inclusive es posible hacer cambios a las configuraciones dinámicamente sin tener un impacto en el tráfico de red. En consecuencia, el tiempo de disponibilidad y fiabilidad se incrementa de manera importante.

Además de elevar la calidad de los servicios de red, HAProxy proporciona una capa extra de seguridad al realizar un filtrado de las peticiones web. El filtrado detecta y previene diferentes tipos de ataques como los de negación de servicio y los de fuerza bruta. Las herramientas de monitoreo permiten estudiar el tráfico y modificar las configuraciones de acuerdo a los comportamientos observados.

Nmap

Nmap (también conocido como Zenmap) es una herramienta de código libre y distribución gratuita para el análisis de redes que te permite auditar y obtener datos de la misma. El *software* usa paquetes IP para descubrir dispositivos, detectar servicios disponibles e incluso detectar el tipo de contrafuegos en uso. Nmap está disponible para los principales sistemas operativos como Windows, Linux y Mac OS [15].

Wireshark

Wireshark es un analizador de red que permite capturar y analizar el tráfico que fluye a través de ella. El software es de código abierto y disponible de manera gratuita en las principales plataformas como Windows, Linux, Mac OS e incluso UNIX. Wireshark es usado por analistas de redes, docentes y académicos debido a su excelente soporte de protocolos de red. Posee la capacidad de capturar paquetes y desplegar su información como origen, destino, contenido y tiempo de vida. Por esta razón es una de las herramientas más poderosas para los expertos en seguridad [16].

6. Desarrollo del Proyecto

6.1 Módulo de Generación de Estadísticas

6.1.1 Recopilación de Datos

El Consejo de Salubridad General provee servicios de acreditación, consultoría y verificación que requieren un constante acceso a su información como dictámenes, formatos y reportes. Gran parte del tráfico de la página web del CSG es interno, por lo tanto es importante conocer la cantidad de tráfico que recibe así como la carga de la red en un día laboral típico.

La recopilación de datos se llevó a cabo en una de las locaciones del consejo utilizando las herramientas de análisis Nmap, Wireshark y como apoyo adicional estadísticas del sitio SimilarWeb [17]. En la Figura 2 muestra el centro de procesamiento de datos desde donde se realizaron las labores de monitoreo y análisis de la red del CSG.

Figura 2. El centro de procesamiento de datos del CSG

Corriendo un escaneo de la red con Nmap, se pudo descubrir cuantos dispositivos están conectados en un día laboral normal. Se incluyen los resultados de la red cableada y la red inalámbrica. Los resultados se muestran a continuación:

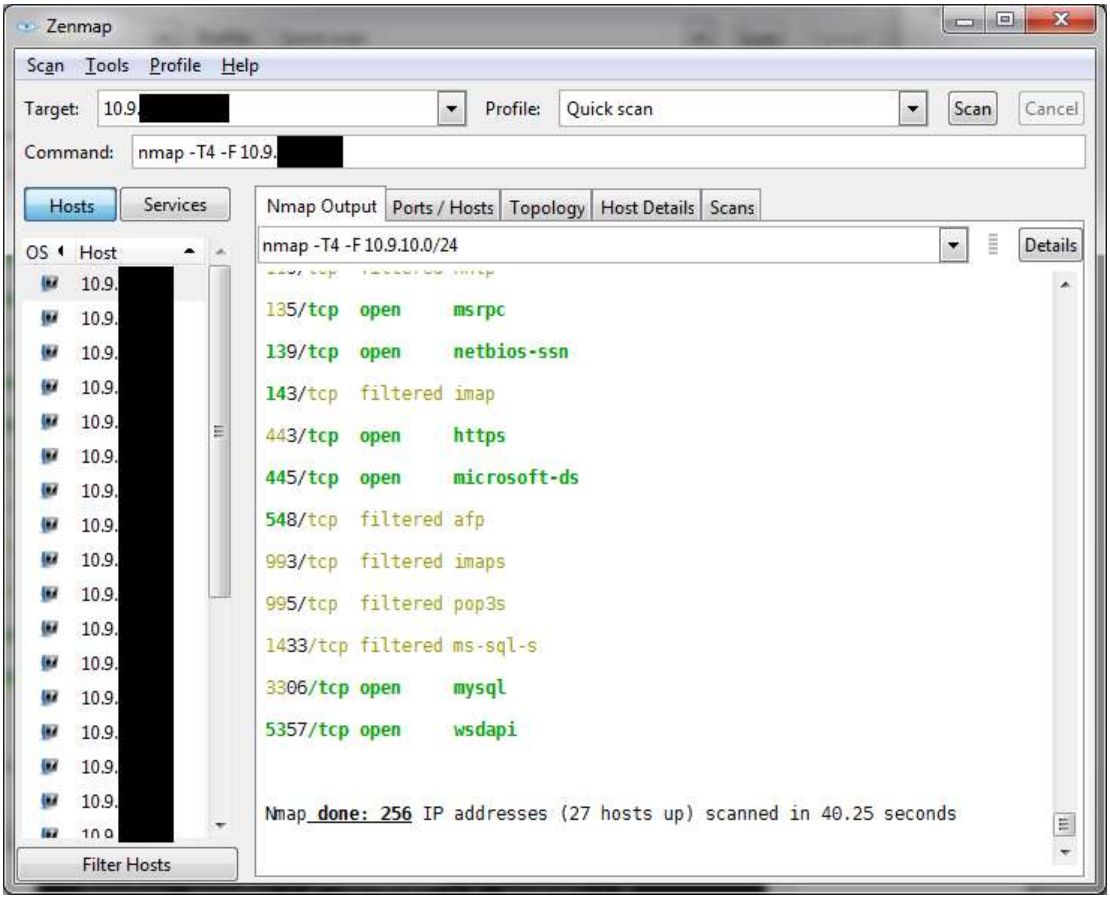


Figura 3. Datos obtenidos por Nmap para la red alámbrica

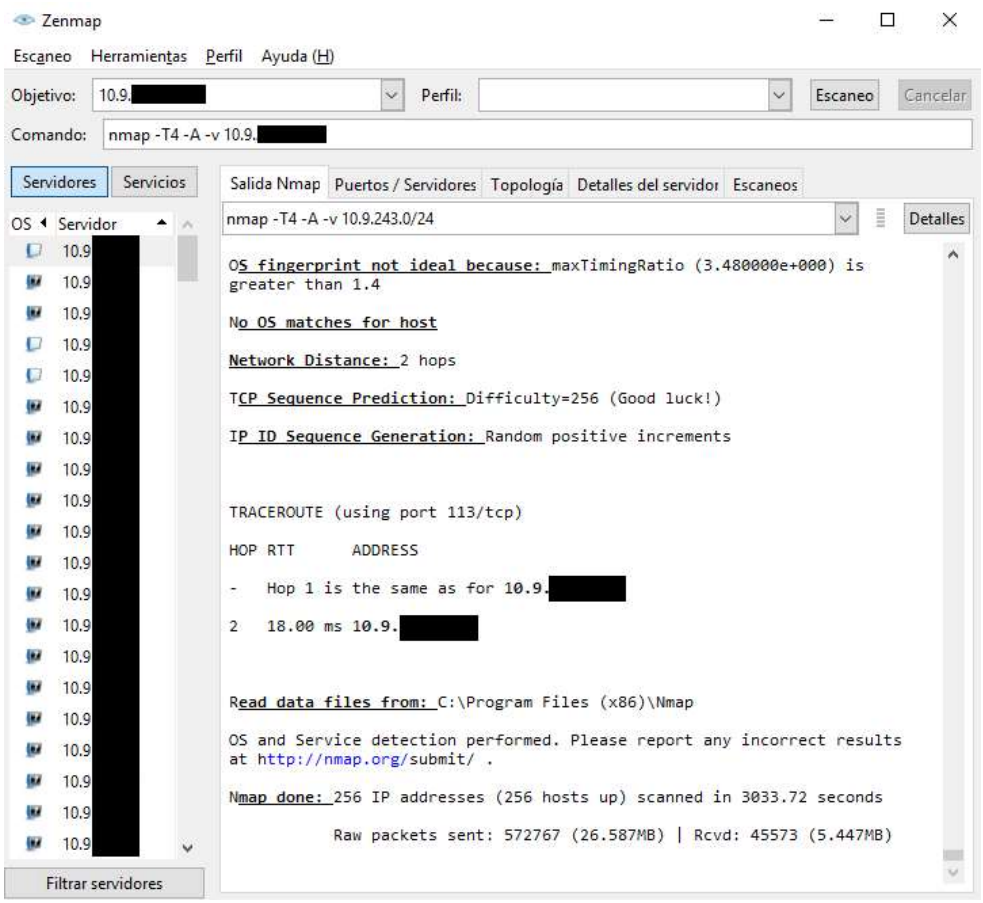


Figura 4. Datos obtenidos por Nmap para la red inalámbrica

En la Figura 3 y 4 se puede observar que se detectaron alrededor de 27 dispositivos activos en la red interna del CSG y 256 dispositivos en su red inalámbrica. Esto solo corresponde a equipos de cómputo ya que los dispositivos de VOIP se encuentran en otra VLAN y estos no accesan al servidor web.

Ahora se procede a analizar con Wireshark. Este análisis capturó el tráfico de la red por alrededor de una hora lo que permitió ver los picos de tráfico. Se encontró que la red local llega a un pico de 1,400,000 bytes/segundo, es decir, 11.2 Megabits/segundo en sus horas de alta productividad.

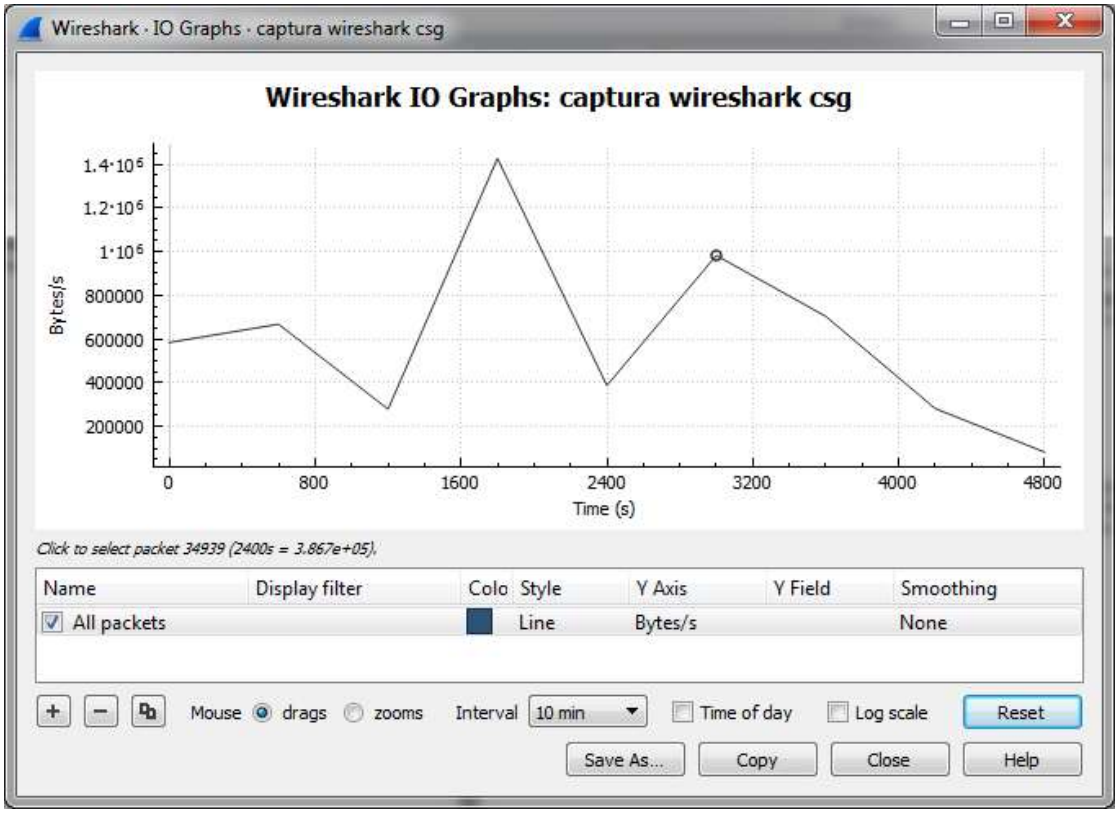


Figura 5. Datos obtenidos por Wireshark

Como información adicional, se consultó el tráfico mensual del sitio web por medio del servicio SimilarWeb. Se utilizó este servicio gratuito ya que la Dirección General de Tecnologías de Información (DGTI), encargada de la página web, no contaba con estos datos.

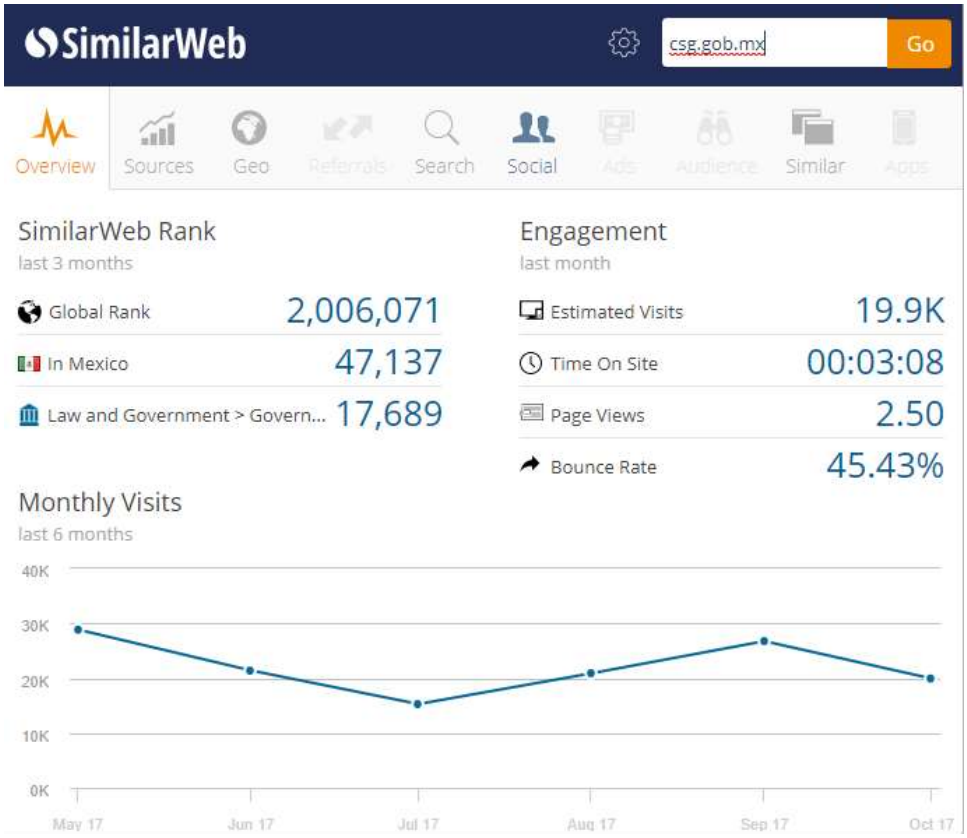


Figura 6. Datos reportados por SimilarWeb

Las estadísticas muestran un máximo de casi 30 mil visitas mensuales, lo cual equivale aproximadamente a un promedio de mil visitas diarias al sitio web. Con los datos recopilados de su red, se realizan las políticas de balance en el siguiente módulo.

6.2 Módulo de Políticas de Balance

6.2.1 Generación de Políticas de Balance

Usando las estadísticas obtenidas en el módulo anterior, se realiza una estimación de rendimiento planeado a futuro y las políticas de balance a utilizar. El director del área de sistemas ha solicitado una holgura de rendimiento de al menos 10% extra para futuros dispositivos y que se use el algoritmo “Leastconn” para el servidor de balance; este algoritmo selecciona el servidor con el menor número de conexiones activas.

El servidor de balance es un IBM xSeries 226 con las siguientes características:

Tarjeta Madre: MSI E7525 Master S2
Procesador: 1x 3GHz Intel Xeon
Chipset: Intel E7525
Memoria: 1GB PC3200
Discos: 2x 73.4GB IBM Ultra320 SCSI
Puertos de Red: 1x Broadcom Gigabit Ethernet
Fuentes de Poder: 2x 514W hot-plug

El servidor ha sido previamente reacondicionado por los técnicos de sistemas y cuenta con el *software* necesario ya instalado (Linux CentOS y HAProxy). En la Figura 7 se muestra el equipo corriendo CentOS 7 de 64 bits.

Figura 7. Servidor de balance corriendo Linux CentOS 7

Las características del balance de carga se encuentran contenidas en el archivo de configuración /etc/haproxy/**haproxy.cfg**, aquí se especifican opciones tales como portal de estadísticas, algoritmo a utilizar y los servidores a balancear.

Existen 4 secciones principales en este archivo de configuración: **global**, **defaults**, **frontend http_front** y **backend http_back**.

En la sección **global** se establecen las configuraciones de estadísticas, un usuario para el *software* de balance y el grupo al que pertenece.

En la sección **defaults** se especifica el protocolo a balancear (HTTP) y los tiempos de inactividad (en ms) permitidos para conexiones, cliente y servidor; además, se especifica una opción para cerrar la conexión del lado del servidor para aumentar un poco el desempeño.

En la sección **frontend http_front** se establece donde correrá el servicio de balance y su dirección de acceso a estadísticas. Otras opciones que se definen en esta parte incluyen características para para evitar abusos:

- "**stick-table**": Aquí se define una tabla que almacena la tasa de conexión de una IP. Su tamaño se definió en 100,000 IPs y expira cada 30 segundos.
- "**tcp-request connection**": Especifica el número máximo de conexiones TCP por cliente. Tan solo se permiten 10 conexiones abiertas por cliente ya que un explorador solo abre un máximo de 7 conexiones para bajar datos. Esto es para evitar abusos y

saturación de ancho de banda. Si el límite es rebasado, el servidor termina la conexión.

- "**acl abuse src_http_req_rate**": Define el número máximo de peticiones por cliente. Similar al parámetro anterior, se estableció un máximo de 10 peticiones. Si el usuario excede el límite, se le etiqueta como abusador y se rechaza las peticiones siguientes.

Por último, la sección **backend http_back** contiene el algoritmo de balance "**leastconn**", la configuración de un archivo temporal *cookie*[5] y las IP de los servidores en el clúster de balance con un máximo de 1000 conexiones cada uno según los datos del análisis.

En la Figura 8 se muestran las configuraciones establecidas. Se han omitido la dirección de las estadísticas y las direcciones IP de los servidores por motivos de seguridad.

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
 GNU nano 2.3.1                Fichero: haproxy.cfg

global
   log /dev/log local0
   log /dev/log local1 notice
   chroot /var/lib/haproxy
   stats socket /var/lib/haproxy/socket/admin.sock mode 660 level admin
   stats timeout 30s
   user haproxy
   group haproxy
   daemon

defaults
   log global
   mode http
   option httplog
   option dontlognull
   timeout http-request 5000
   timeout connect 5000
   timeout client 10000
   option http-server-close
   timeout server 10000

frontend http_front
   bind 0.0.0.0:80
   stats uri [redacted]
   default_backend http_back

  stick-table type ip size 100k expire 30s store conn_rate(3s)

  #IPs limpias pueden pasar el filtro
   tcp-request connection accept if { src -f /etc/haproxy/whitelist.lst }
  #Cerrar conexion si cliente tiene 10 abiertas
   tcp-request connection reject if { src_conn_rate ge 10 }
   tcp-request connection track-scl src

  #Si hay demasiadas peticiones se etiqueta al abusador
    acl abuse src_http_req_rate(http_front) ge 10
    acl flag_abuser src_inc_gpc0(http_front) ge 0
    tcp-request content reject if abuse flag_abuser


backend http_back
   balance leastconn
   cookie MYSRV insert indirect nocache
   server server1 [redacted] check cookie srv1 maxconn 1000
   server server2 [redacted] check cookie srv2 maxconn 1000

                              [ 48 líneas leídas ]
^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y Pág Ant      ^K CortarTxt    ^C Pos actual
^X Salir          ^J Justificar   ^W Buscar       ^V Pág Sig      ^U PegarTxt     ^T Ortografía
```

Figura 8. Parámetros de configuración de balance de carga

[5] Son archivos de texto que se descargan en un ordenador y permiten almacenar datos de navegación.

6.2.2 Generación de Script de inicio

En lo que respecta a la configuración de seguridad y rendimiento en contra de ataques DDoS, esta se encuentra en un *script* del servicio **sysctl** que modifica opciones del *kernel*[6] de Linux (código básico del sistema operativo). El archivo de configuración /etc/**sysctl.conf** contiene las características que controlan ciertos parámetros de Linux.

Los parámetros más importantes son los siguientes:

- **"net.ipv4.tcp_syncookies"**: Se activan las cookies SYN que ayudan a mitigar ataques por inundación.
- **"net.ipv4.conf.all.rp_filter"**: Se activa el filtro rp para poder encaminar paquetes por la interface en la que entro.
- **"net.ipv4.tcp_max_syn_backlog"**: Se utiliza para configurar el número máximo de conexiones TCP esperando confirmación. Se estableció un límite de 1024 basados en el análisis previo.
- **"net.core.rmem_max**": Se incrementa el tamaño máximo de buffer a recibir.
- **"net.core.wmem_max**": Se incrementa el tamaño máximo de buffer a enviar.
- **"net.ipv4.tcp_max_tw_buckets":**Determina el número máximo de conexiones Time Wait
- **"net.ipv4.tcp_tw_reuse":** Permite reusar conexiones Time Wait para nuevas peticiones.

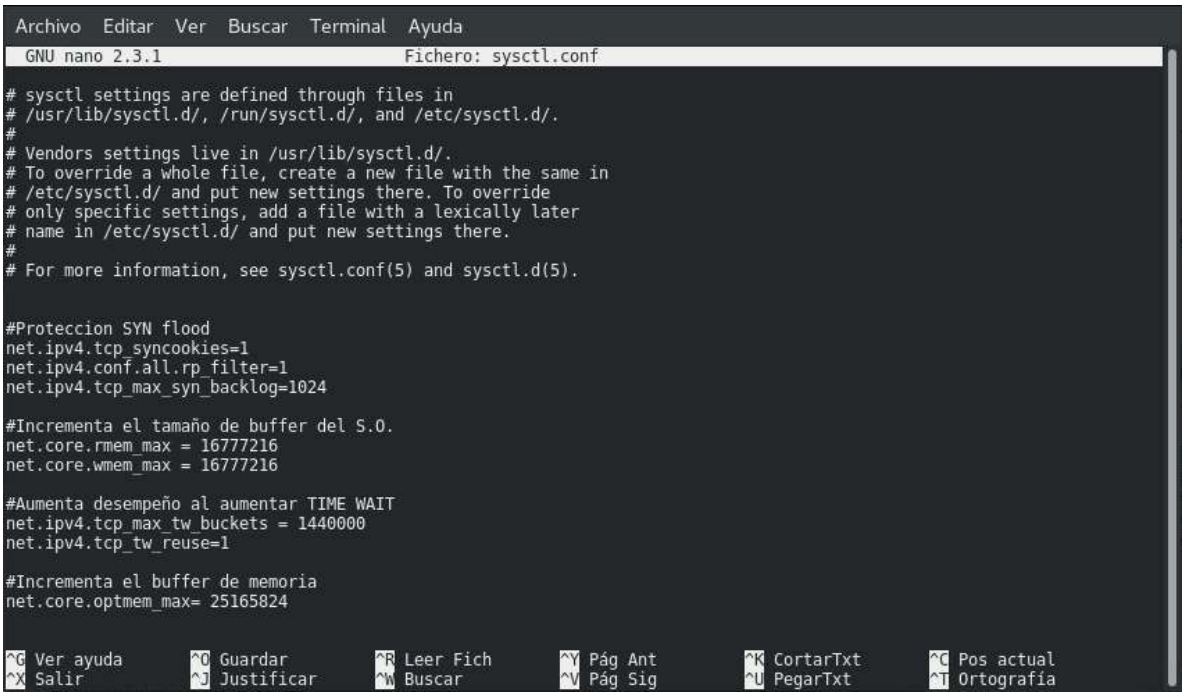


Figura 9. Script de configuración de seguridad en sysctl

[6] Es la parte del sistema operativo que funciona a más bajo nivel y se encarga de manejar la comunicación con el hardware, la integridad y seguridad de todo el sistema.

Una vez establecidas las configuraciones, se procede a realizar las pruebas y su respectiva retroalimentación en el siguiente módulo.

6.2.3 Configuración de Acceso Remoto

Linux CentOS tiene configurado por defecto el acceso remoto por SSH[7], sin embargo se desea agregar un banner de bienvenida cada que un usuario ingrese al servidor de balance. Para esto, es necesario modificar el archivo de configuración **/etc/ssh/sshd_config** activando la opción **PrintMotd** para que se imprima el mensaje del día. Posteriormente se especifica el mensaje a mostrar en **/src/motd**

6.2 Módulo de Acreditación de Software

6.3.1 Pruebas de Software

Se realizan pruebas de las configuraciones con las imágenes de los servidores a balancear, corriendo en máquinas virtuales con IP estáticas dentro de la misma red del CSG. Esto es para simular, de la manera más certera, el comportamiento del software y para no interrumpir el servicio de la página web del CSG cuando se ejecute cada prueba.

Primero, se verifica que el servidor sea accesible de manera remota mediante SSH, se ingresa el IP, usuario y contraseña para recibir el acceso.

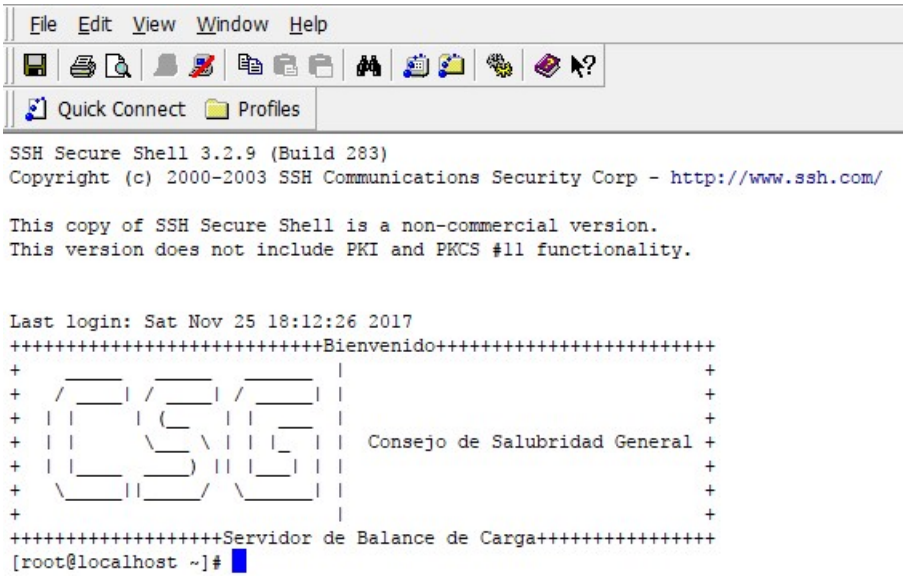


Figura 10. Conexión remota SSH al servidor de balance

[7] SSH (*Secure Shell*) es un protocolo de comunicación para realizar conexiones remotas usando encriptación de 128 bits.

Ahora se verifica el funcionamiento del software de balance. En la Figura 11 se muestra que el servicio de balance se está ejecutando de manera apropiada. Si existiera algún error en el archivo de configuración, el servicio se detendría.

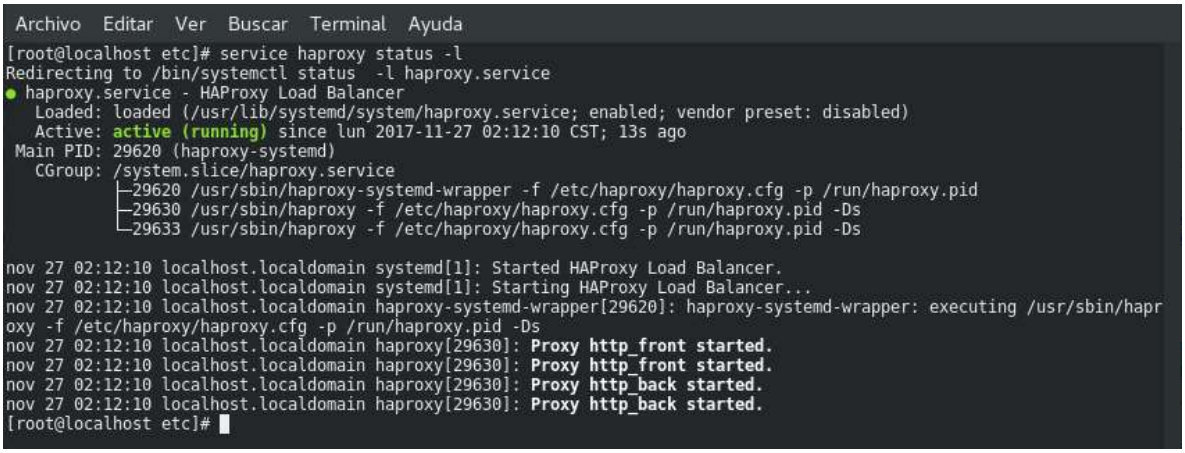


Figura 11. El servicio de balance inició correctamente y se encuentra en ejecución

También se verifica que las configuraciones del script **sysctl** se hayan cargado de manera correcta y que no existan errores o conflictos.

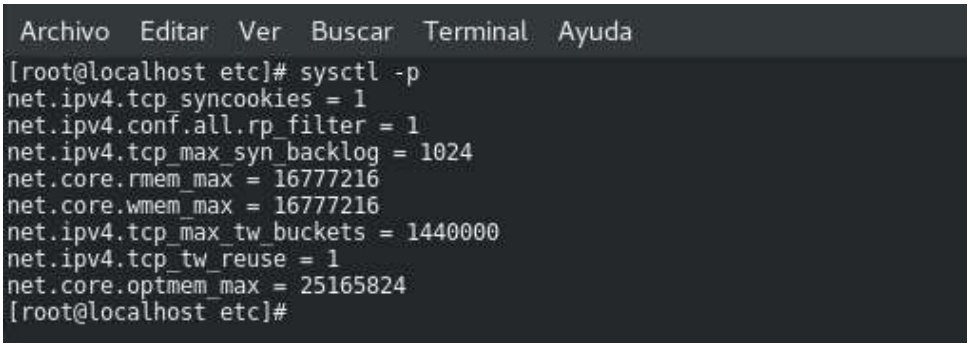
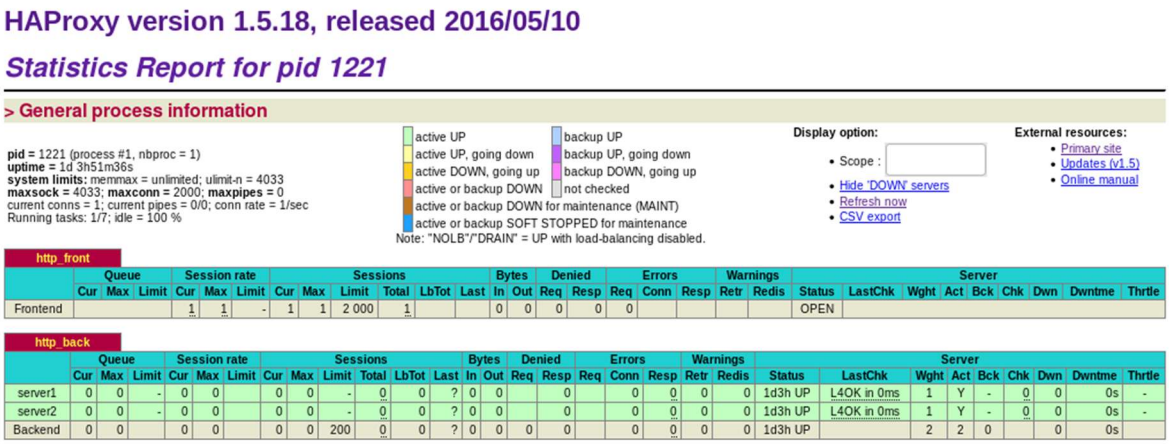


Figura 12. Reporte de sysctl no muestra errores de configuración

Ya que se montaron las imágenes de los servidores en máquinas virtuales dentro de la misma red del CSG. Se configuran sus IP dentro del software de balance; posteriormente, se procede a verificar su funcionamiento en el portal de estadísticas de HAProxy.



HAProxy version 1.5.18, released 2016/05/10

Statistics Report for pid 1221

> General process information

pid = 1221 (process #1, nbproc = 1)
uptime = 1d 3h51m36s
system limits: memmax = unlimited; ulimit-n = 4033
maxsock = 4033; **maxconn** = 2000; **maxpipes** = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec
Running tasks: 1/7; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
backup UP
backup UP, going down
backup DOWN, going up
not checked
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide 'DOWN' servers
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v1.5)
- Online manual

http_front

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	1	-	1	1	2 000	1			0	0	0	0	0					OPEN								

http_back

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
server1	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	1d3h UP	L4OK in 0ms	1	Y	-	0	0	0s	-
server2	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	1d3h UP	L4OK in 0ms	1	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0	200	0	0	?	0	0	0	0		0	0	0	0	1d3h UP		2	2	0		0	0s	

Figura 13. Portal de estadísticas de HAProxy

Se inician las pruebas desde otro equipo de apoyo corriendo Linux y se verifica en el portal de estadísticas que todas las peticiones sean redireccionadas de manera correcta. Para empezar, se realizan 30000 peticiones con 100 conexiones concurrentes (Ya que las políticas de balance solo permiten 10 conexiones por usuario, esta opción ha sido modificada temporalmente para estas pruebas). Esto simula la carga que tendría la página web durante el mes más ocupado.

Se utiliza el comando de Apache Bench con los siguientes parámetros:

ab -n 30000 –c 100 <Destino>

Donde **n** es el número de peticiones totales, **c** es el número de conexiones concurrentes y su respectivo destino, ya sea una IP o una dirección URL. Los resultados se muestran a continuación en la Figura 14 y 15.

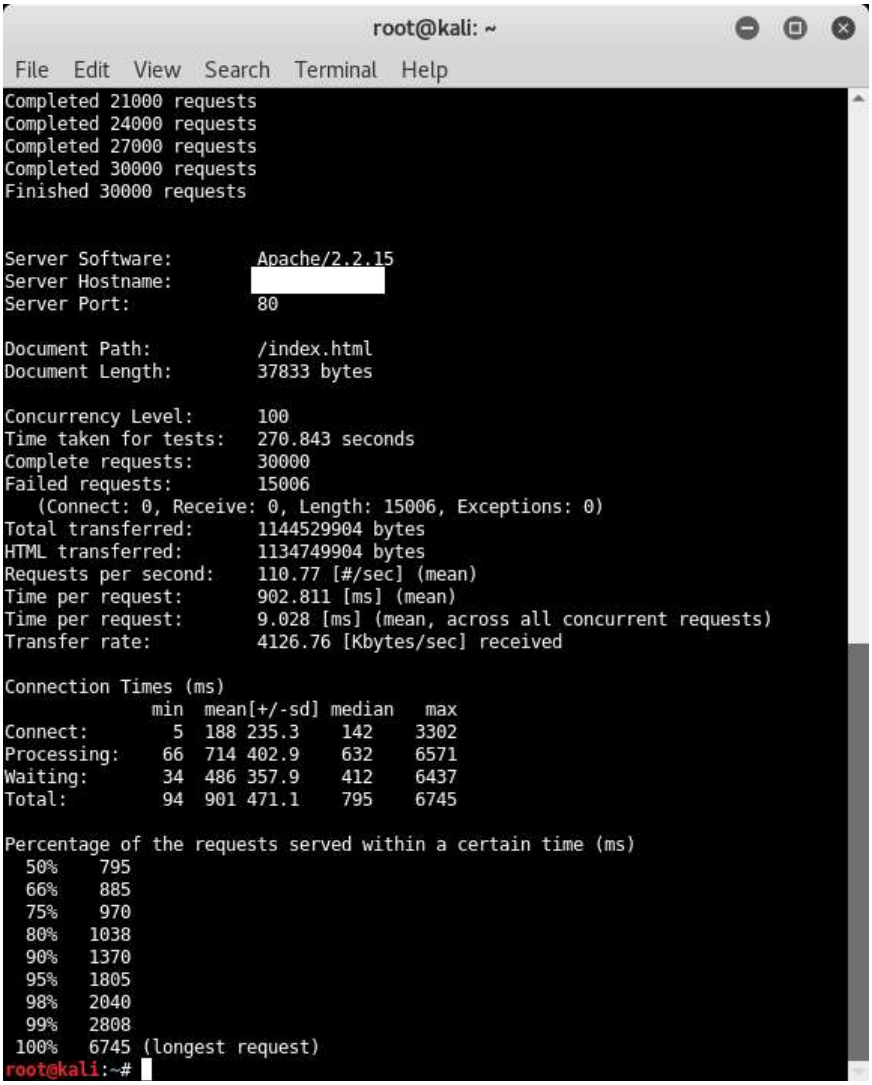


Figura 14. Peticiones realizadas desde el equipo de apoyo

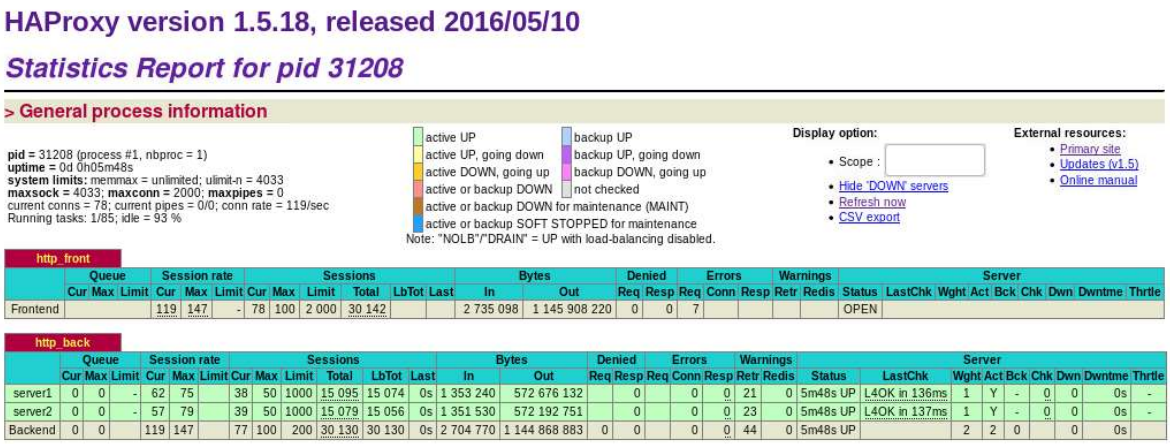


Figura 15. Las estadísticas muestran el funcionamiento del software de balance después de la simulación

Ahora se procede a verificar el comportamiento del servidor con un número de peticiones HTTP aun mayor, aumentando el número de conexiones concurrentes a 2000. Esto simularía un posible ataque DoS.

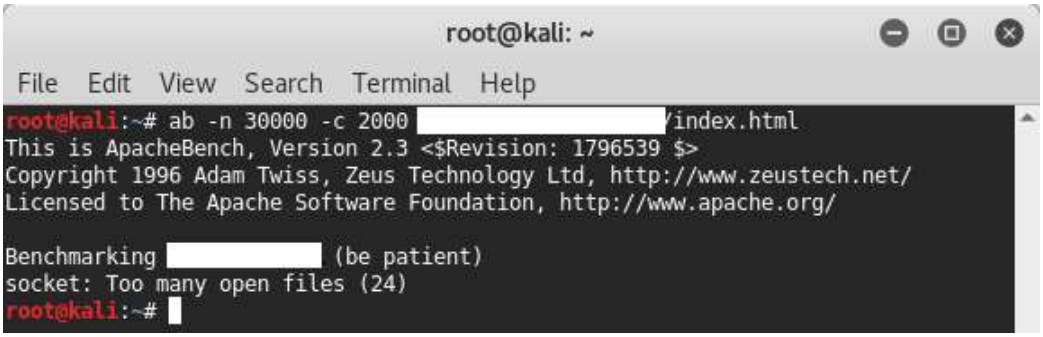


Figura 16. La petición es rechazada por el alto número de archivos index abiertos

Incluso, si se reduce el número de peticiones al máximo permitido (1000 para cada servidor web), el servidor de balance maneja correctamente la carga que se le presenta al cerrar la conexión.

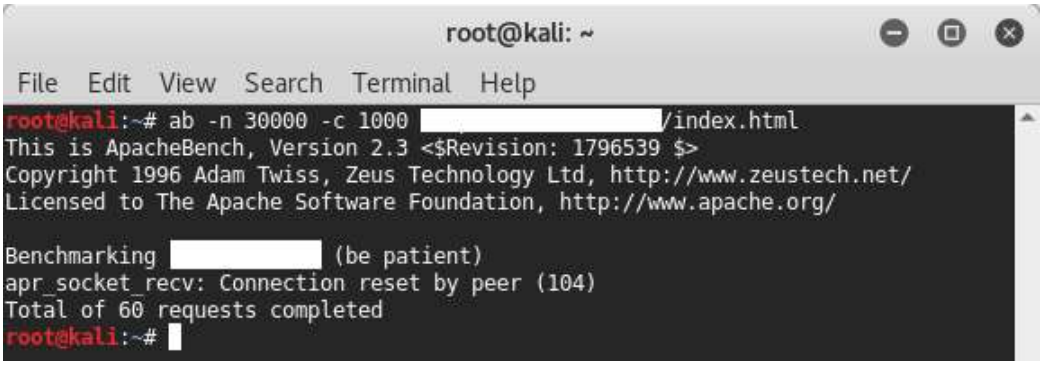
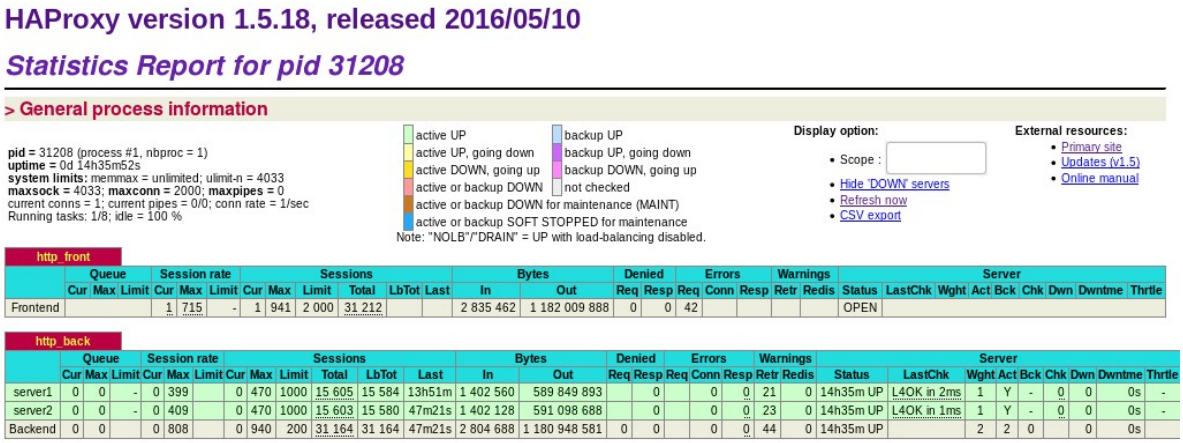


Figura 17. El servidor cierra la conexión al detectar el abuso.

Se verifica en el portal de estadísticas que las peticiones no hayan sido completadas y se puede ver en la sección **Errors - Req** del **http_frontend.**



HAProxy version 1.5.18, released 2016/05/10

Statistics Report for pid 31208

> General process information

pid = 31208 (process #1, nbproc = 1)
uptime = 0d 14h35m52s
system limits: memmax = unlimited; ulimit-n = 4033
maxsock = 4033; maxconn = 2000; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec
Running tasks: 1/8; idle = 100 %

active UP | backup UP
active UP, going down | backup UP, going down
active DOWN, going up | backup DOWN, going up
active or backup DOWN | not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
• Scope :
• Hide 'DOWN' servers
• Refresh now
• CSV export

External resources:
• Primary site
• Updates (v1.5)
• Online manual

http_front

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	715	-	1	941	2 000	31 212			2 835 462	1 182 009 888	0	0	42					OPEN								

http_back

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
server1	0	0	-	0	399		0	470	1000	15 605	15 584	13h51m	1 402 560	589 849 893		0		0	0	21	0	14h35m UP	L4OK in 2ms	1	Y	-	0	0	0s	-
server2	0	0	-	0	409		0	470	1000	15 603	15 580	47m21s	1 402 128	591 098 688		0		0	0	23	0	14h35m UP	L4OK in 1ms	1	Y	-	0	0	0s	-
Backend	0	0		0	808		0	940	200	31 164	31 164	47m21s	2 804 688	1 180 948 581	0	0		0	0	44	0	14h35m UP		2	2	0		0	0s	

Figura 18. Reporte de HAProxy después de los “ataques”

Con esto se comprueba que el filtrado de las peticiones se realiza de manera correcta desde el *front end* del servidor y las peticiones maliciosas no llegan hasta el clúster de balance en el *back end*. Los usuarios legítimos no se ven afectados.

La última prueba consiste en el rendimiento extra solicitado por el Director de Sistemas. Si el número máximo de visitas en un mes es de 30,000, un 10% extra corresponde a 3,000 visitas más; sin embargo, se realiza una prueba con un 15% más, es decir, un total de 34,500 peticiones con 150 concurrentes.

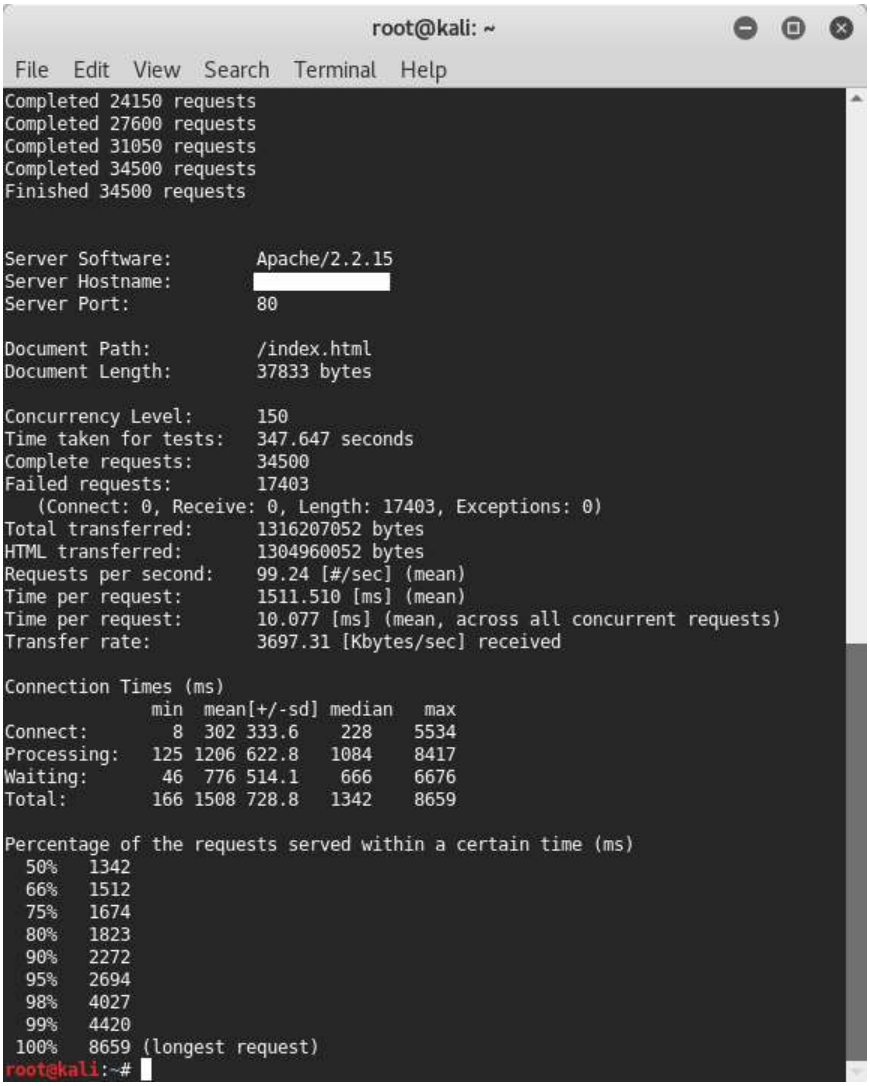
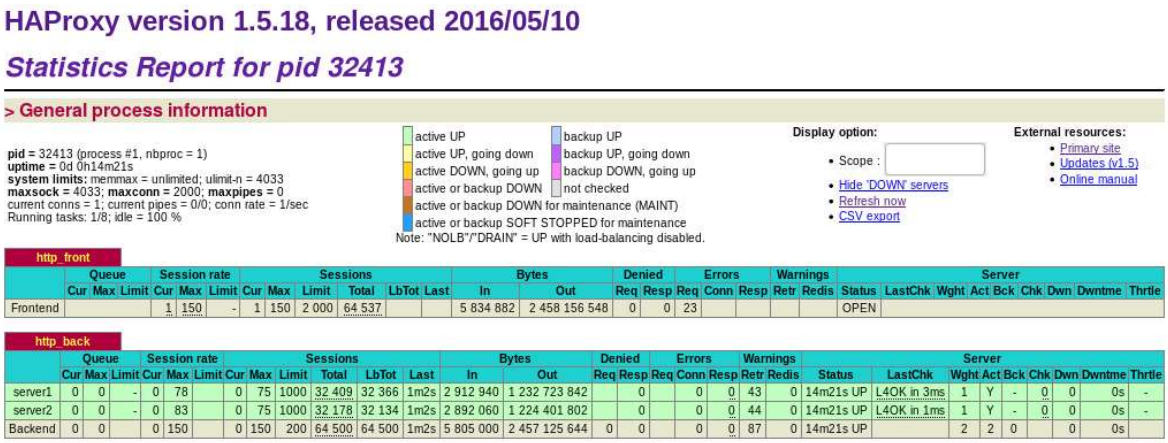


Figura 19. Peticiones de rendimiento



HAProxy version 1.5.18, released 2016/05/10

Statistics Report for pid 32413

> General process information

pid = 32413 (process #1, nbproc = 1)
uptime = 0d 0h14m21s
system limits: memmax = unlimited; ulimit-n = 4033
maxsock = 4033; **maxconn** = 2000; **maxpipes** = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec
Running tasks: 1/8; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
backup UP
backup UP, going down
backup DOWN, going up
not checked
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide 'DOWN' servers
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v1.5)
- Online manual

http_front

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	150	-	1	150	2 000	64 537			5 834 882	2 458 156 548	0	0	23					OPEN								

http_back

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
server1	0	0	-	0	78		0	75	1000	32 409	32 366	1m2s	2 912 940	1 232 723 842		0		0	0	43	0	14m21s UP	L4OK in 3ms	1	Y	-	0	0	0s	-
server2	0	0	-	0	83		0	75	1000	32 178	32 134	1m2s	2 892 060	1 224 401 802		0		0	0	44	0	14m21s UP	L4OK in 1ms	1	Y	-	0	0	0s	-
Backend	0	0		0	150		0	150	200	64 500	64 500	1m2s	5 805 000	2 457 125 644	0	0		0	0	87	0	14m21s UP		2	2	0		0	0s	

Figura 20. Reporte de HAProxy después la prueba de rendimiento extra

7. Resultados

El servidor de balance cumplió con el objetivo principal de este proyecto al administrar de manera satisfactoria los diferentes niveles de carga que se le presentaron. Se comprobó que puede proteger ante ataques de inundación como DoS y DDoS por lo que proporciona un nivel extra de seguridad además del contrafuegos ya existente en la red.

El Consejo de Salubridad General no tendrá inconveniente en proveer acceso a su página web tanto a sus funcionarios y al público en general. Además, es posible incrementar el tamaño del clúster de manera sencilla, lo que permitiría aumentar los niveles de tolerancia y carga.

La administración remota permite cambiar las configuraciones de una manera simple sin necesidad de estar físicamente en el centro de procesamiento.

Por último, el servidor soportará el crecimiento solicitado por el Director del área de Sistemas usando la configuración actual sin ninguna modificación adicional.

8. Análisis de Resultados

El análisis inicial requirió de más tiempo del especificado por motivos externos al CSG. Específicamente porque se realizaron labores de mantenimiento a la infraestructura del Consejo. Una vez que se tuvo acceso al centro de datos se corrieron las herramientas de red y se hizo un censo sobre los hábitos de uso de sus usuarios. Esto nos arrojó que el acceso al servidor web es relativamente bajo pero constante, por lo era preciso cubrir los requerimientos de carga en la configuración del servidor de balance.

Se encontró que el servidor puede soportar perfectamente la carga actual y el 10% extra requerido por la Dirección de Sistemas. Sin embargo, la latencia de los paquetes se incrementó bastante, esto a pesar de las optimizaciones realizadas al *kernel* de Linux.

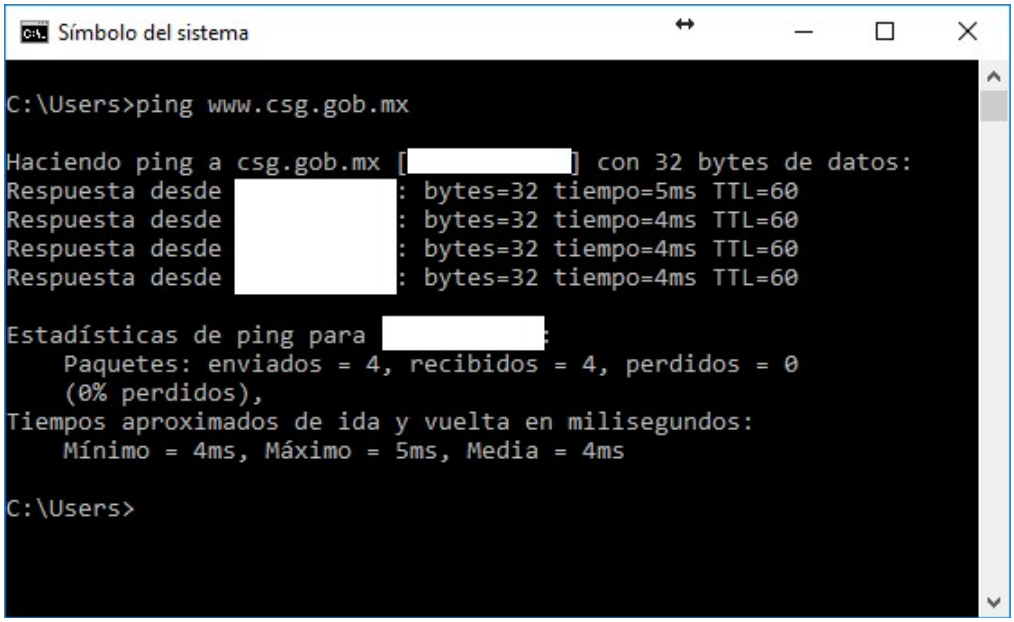


Figura 21. Latencia sin servidor de balance (configuración actual)

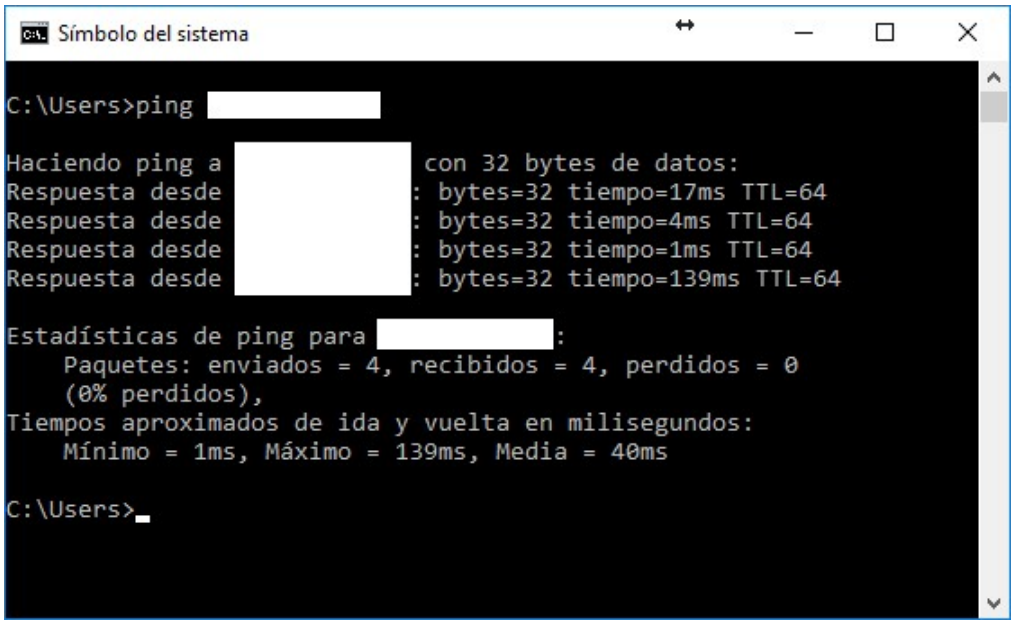


Figura 22. Latencia con servidor de balance

Todo servicio de balance de carga agrega una latencia (retraso) al enrutamiento de paquetes ya que el servicio de balance realiza un análisis de los paquetes y determina la acción apropiada. A pesar de esto, la latencia añadida es imperceptible para los usuarios ya que los tiempos se incrementaron en algunos milisegundos. Se sugirió a la Dirección de Sistemas agregar más memoria RAM al servidor de balance en caso que se aumente el tamaño del clúster de balance.

Durante las pruebas, también se detectó un número de "*Failed requests*". Después de analizar la documentación de Apache, se descubrió que solo se refiere a que la mitad del tiempo, las respuestas tuvieron una longitud diferente. Esto no significa que las peticiones fueron rechazadas o que no llegaron a su destino, y que es perfectamente normal con contenido web dinámico.

Si bien la latencia añadida puede parecer una desventaja para los servicios web, la seguridad también se incrementa. Aparte de esta situación, el proyecto se implementó de manera satisfactoria y cumplió con las expectativas de funcionamiento.

9. Conclusiones

La realización de este proyecto me ha permitido reafirmar mis conocimientos de redes y buscar la solución óptima con los recursos disponibles. La implementación del servidor de balance responde a las necesidades de las redes modernas, y no solo del CSG, donde la alta disponibilidad y seguridad se han vuelto parte fundamental de los servicios de red.

El computo en la nube, los sistemas distribuidos, las bases de datos y la creciente demanda de servicios web en el internet de las cosas, hacen que los servidores de balance, contrafuegos, y otros dispositivos de protección tomen un nuevo peso en la infraestructura de redes. Los centros de datos no solo deben crecer en su capacidad de almacenamiento y transmisión de información, sino que también deben asegurar la integridad de esos datos y la accesibilidad de los mismos.

10. Referencias Bibliográficas

[1] K. Erciyes and S. Yilmaz, "Dynamic load balancing in a distributed computer system," [1991 Proceedings] 6th Mediterranean Electrotechnical Conference, LJubljana, 1991, pp. 1077-1080 vol.2 [En línea]. Disponible: http://ieeexplore.ieee.org

[2] M. Rahman, S. Iqbal and J. Gao, "Load Balancer as a Service in Cloud Computing," 2014 IEEE 8th International Symposium on Service Oriented System Engineering, Oxford, 2014, pp. 204-211.doi: 10.1109/SOSE.2014.31 [En línea]. Disponible: http://ieeexplore.ieee.org

[3] A. Kumar and M. Kalra, "Load balancing in cloud data center using modified active monitoring load balancer," 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Spring), Dehradun, 2016, pp. 1-5. doi: 10.1109/ICACCA.2016.757890 [En línea]. Disponible: http://ieeexplore.ieee.org

[4] "HAProxy - The Reliable, High Performance TCP/HTTP Load Balancer", Haproxy.org, 2017. [En línea]. Disponible: http://www.haproxy.org/

[5] Christos Douligeris, Aikaterini Mitrokotsa, “DDoS attacks and defense mechanisms: classification and state-of-the-art”, Computer Networks, Volume 44, Issue 5, 2004, pp 643-666, ISSN 1389-1286 [En línea]. Disponible: http://www.sciencedirect.com

[6] J. Santana Santana, “Algoritmos de Balance de Carga con Manejo de Información Parcial”, Proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Iztapalapa, México, 2010.

[7] D. Avila Castillo,“Análisis y Gestión de Recursos para Brindar Seguridad en una Red Empresarial”, Proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2015.

[8] "Misión y Visión", Consejo de Salubridad General, 2017. [En línea]. Disponible: http://www.csg.gob.mx/

[9] Andrew S. Tanenbaum y David J. Wetherall, “Redes de Computadoras”, Quinta Edición, Pearson Educacion, 2012

[10] “Introducción al equilibrio de carga de red”, Microsoft Developer Network, 2017, [En línea]. Disponible: https://msdn.microsoft.com

[11] P. Robichaux,”Distributed Denial-of-Service Attacks and You”, Microsoft Developer Network, 2017, [En línea]. Disponible: https://msdn.microsoft.com

[12] Schneiner Bruce, “BOTNETS of Things”, MIT Technology Review, Mar/Apr2017, Vol. 120 Issue 2 [En linea] Disponible: https://www.ebsco.com

[13] "About CentOS", Centos.org, 2017. [En línea]. Disponible: https://www.centos.org/

[14] “Operating system Share”, TOP500.org, 2017, [En línea]. Disponible: https://www.top500.org/statistics/list/

[15] “Nmap: the Network Mapper - Free Security Scanner”, Nmap.org, 2017. [En línea] Disponible: https://nmap.org/

[16] “Wireshark”, wireshark.org, 2017 [En línea] Disponible: http://www.wireshark.org/

[17] “About SimilarWeb”, similarweb.com, 2017 [En línea] Disponible: http://www.similarweb.com/