



02 de julio de 2021
Ficha N° 10 TSHARK de Comandos
CSIRT DE GOBIERNO

Comando de la semana “TSHARK”

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 Scripting¹, 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 múltiples oportunidades los analistas de ciberseguridad debemos recurrir a analizar el tráfico de red, es decir, lo que está fluyendo por los cables de red o canales inalámbricos, para identificar anomalías o para confirmar que las comunicaciones se están produciendo al menos a nivel de capa de red y que los problemas podrían estar aguas arriba en la capa de aplicación.

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 mitigación²,

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>



monitoreo y vigilancia. Como bien sabemos, los tres ejes básicos de la ciberseguridad son la Confidencialidad, la Integridad y la Disponibilidad. En esta oportunidad hablaremos de una herramienta que puede afectar la confidencialidad de las comunicaciones, razón por la cual es muy importante que nuestros analistas estén adscrito y bajo el paragua de una muy buen apolítica de seguridad de la información, un código de ética suficientemente robusto para guiar de buena manera su actuar y un NDA (o acuerdo de no divulgación) que deje clara las reglas sobre la información sensible que pueda tomar conocimiento durante su trabajo.

La herramienta TSHARK al igual que muchas otras³ nos dan visualización de lo que fluye por la red, y por tanto es vital entender ese idioma para poder leer lo que se conversa entre máquinas. Indudablemente hay que dejar en claro que las comunicaciones que van cifradas, efectivamente no se pueden abrir o leer a menos que utilicen un protocolo de cifrado muy débil o que se haya intervenido la red para colocar un dispositivo intermedio que abra y cierre el cifrado entre el cliente y el servidor. Todas las comunicaciones que fluyan sin cifrado quedarán a la vista del analista, y por tanto se deben tomar los resguardos de seguridad para trabajar con los datos que se puedan visualizar de estos análisis, en coherencia con los marcos regulatorios internos y la legislación vigente sobre la materia.

¿Qué es TSHARK?

Primero brevemente responderemos a la pregunta ¿Qué es un sniffer?:

Un Sniffer o analizar de red, son utilidades tanto de hardware o de software que se ha desarrollado con el objetivo de generar un monitoreo constante del tráfico de la red local o externa. Este rastreo básicamente se encarga de analizar los flujos de paquetes de datos que son enviados y recibidos entre los equipos de la red ya sea a nivel interno o externo.

Se hace uso de un modo de rastreo llamado "modo promiscuo" con el cual nos da la oportunidad de examinar todos los paquetes sin importar su destino, esto puede tomar tiempo pero es clave para saber con certeza que pasa por nuestra red.

Un Sniffer lo podemos configurar de dos modos diferentes, dependiendo del requerimiento de soporte, estos modos son:

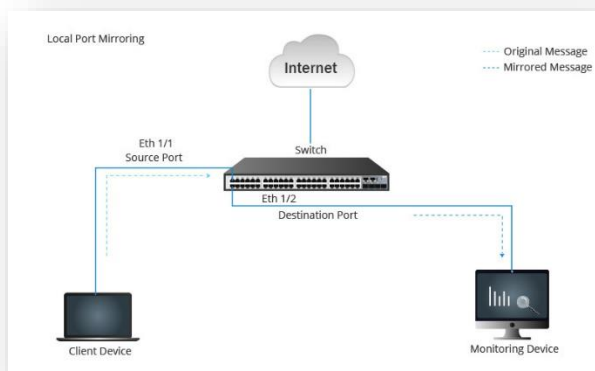
- Lo podemos configurar sin filtro para que la herramienta capture todos los paquetes disponibles y almacene un registro de ellos en el disco duro local con el fin de analizarlos más adelante.

³ Wireshark, Microsoft Message Analyzer, Tcpdump, Windump, Lizard Systems WiFi Scanner, Capsa Network Analyzer, Netcat, InnoNWSniffer, SniffPass, EtherApe, Kismet, NetworkMiner entre otros.



- Puede ser configurado con un filtro específico lo cual nos da la oportunidad de capturar paquetes en base a los criterios que especifiquemos antes de la búsqueda

Los Sniffers o analizadores de red podemos usarlos por igual en una red LAN o Wi-Fi. La principal diferencia radica en que si este se usa en una red LAN, tendremos acceso a los paquetes de cualquier equipo conectado (estudie los conceptos de “Port SPAN”⁴ y “Port Mirror”⁵ y sus multiples configuraciones en los equipos switch). O bien se puede establecer un limitante en base a los dispositivo de red, en el caso de usarse en una red inalámbrica, el analizador de red sólo podrá escanear un canal a la vez por la limitante de la red, pero si hacemos uso de varias interfaces inalámbricas esto puede mejorar un poco, pero siempre será mejor usarse en una red cableada o LAN.



Cuando rastreamos los paquetes usando un Sniffer o un analizador de red, podemos acceder a detalles como (estudie las aplicaciones y los protocolos inseguros que utilizan⁶):

- Información de los sitios visitados
- Contenido y destinatario de los correos electrónicos enviados y recibidos
- Ver archivos descargados y muchos más detalles
- Usuarios y contraseñas de acceso a sistemas inseguros
- Consultas de monitoreo snmp inseguras y las comunidades utilizadas

El objetivo fundamental de un Sniffer es analizar todos los paquetes de la red, especialmente del tráfico entrante, para buscar cualquier objeto cuyo contenido integre código malicioso y de esta forma aumentar la seguridad en la organización evitando que se instale en cualquier equipo cliente algún tipo de malware.

⁴ https://es.wikipedia.org/wiki/Puerto_espejo

⁵ <https://community.fs.com/es/blog/port-mirroring-explained-basis-configuration-and-fa-qs.html>

⁶ Básicamente, cualquier protocolo que no brinde autenticidad, integridad y confidencialidad. En la práctica, esto significa que HTTP debería ser HTTPS, FTP debería ser FTPS o SFTP, telnet debería ser SSH, POP3 debería ser POP3S e IMAP debería ser IMAPS. Es importante deshabilitar todos los demás protocolos y no solo proporcionar una alternativa segura. El protocolo encriptado debe proporcionar fuerte criptografía. Cualquier comunicación que envíe datos confidenciales del titular de la tarjeta a través de la red pública debe usar un canal cifrado como SSL/TLS o, por ejemplo, IPSEC.



TSHARK

Como lo que estamos fomentando con estas fichas es el uso de comandos de consola, para facilitar posteriores tareas y playbooks de automatización dentro del quehacer de un analista de ciberseguridad nos concentraremos en TSHARK, pero no podemos dejar de mencionar de donde proviene: WIRESHARK, que es la versión gráfica de nuestro sniffer y cuyas propiedades están incluidas también en TSHARK.

Tshark es un analizador de protocolos de red. Permite capturar paquetes de datos desde una red en vivo, o leer los paquetes desde un archivo de captura previamente guardado, o imprimir un formulario descifrado los paquetes o escribir los paquetes en un archivo.

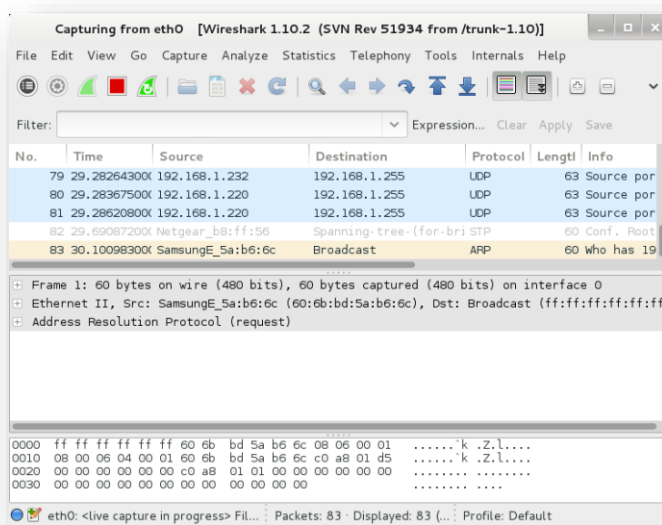


Ilustración 1: Wireshark

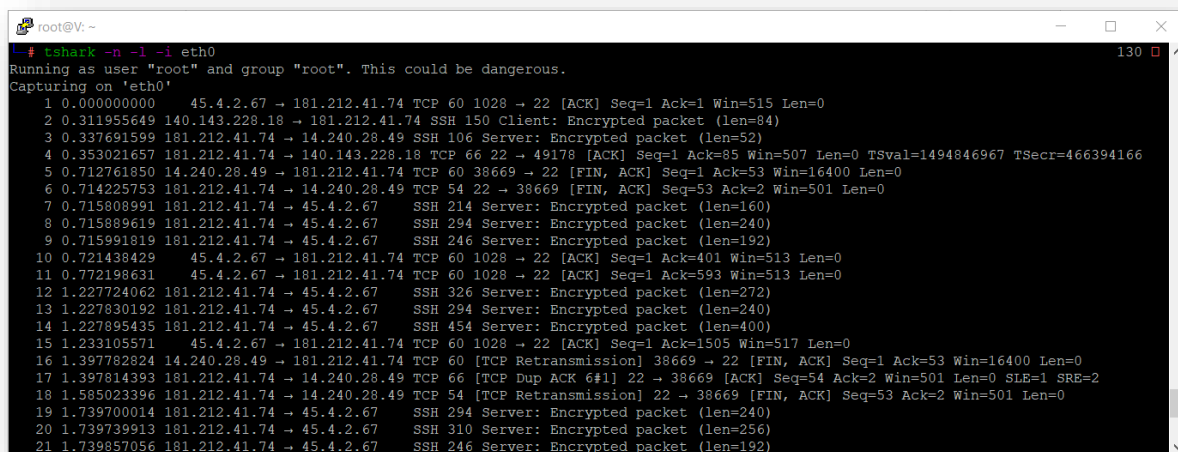


Ilustración 2: TSHARK

TSHARK tiene un rico conjunto de funciones que incluye lo siguiente:

-  Inspección profunda de cientos de protocolos.



- Captura en vivo y análisis fuera de línea.
- Los datos de red capturados se pueden navegar en modo TTY
- Los filtros de pantalla más potentes de la industria
- Análisis completo de VoIP
- Los archivos de captura comprimidos con gzip se pueden descomprimir sobre la marcha
- Los datos en vivo se pueden leer desde Ethernet, IEEE 802.11, PPP / HDLC, ATM, Bluetooth, USB, Token Ring, Frame Relay, FDDI y otros (según su plataforma)
- Soporte de descifrado para muchos protocolos, incluidos IPsec, ISAKMP, Kerberos, SNMPv3, SSL / TLS, WEP y WPA / WPA2

NOTA IMPORTANTE: Dado que es relevante un buen manejo de los comandos básicos de Linux, tanto para posteriores manejos de los datos o archivos 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.

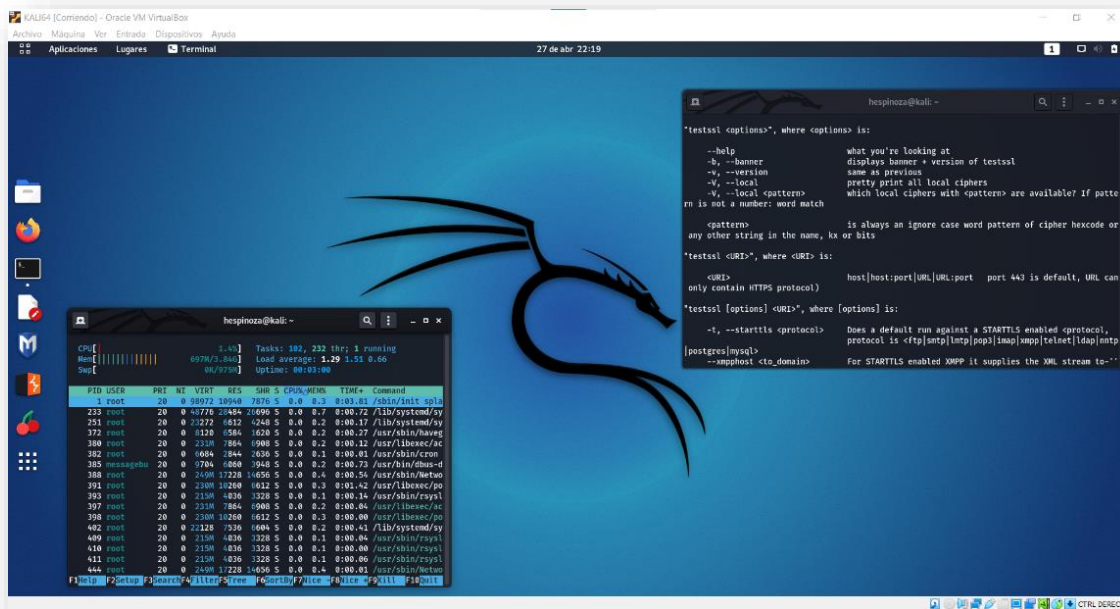
Vea anexo I: Comandos básicos de Linux



III. PASO A PASO

PASO 1: Un entorno adecuado para trabajar.

Primero debe contar con una distribución de Kali⁷ Linux funcionando ya sea en una máquina física o en una máquina virtual⁸⁹.



Instalación de Kali Linux

La instalación de Kali Linux (arranque único) en su computadora es un proceso sencillo. Esta guía cubrirá la instalación básica (que se puede realizar en una máquina virtual invitada o sobre un equipo entero), con la opción de cifrar la partición. En ocasiones, es posible que tenga datos confidenciales que preferiría cifrar con Full Disk Encryption (FDE). Durante el proceso de instalación, puede iniciar una instalación cifrada LVM en el disco duro o en las unidades USB.

Primero, necesitará hardware de computadora compatible. Kali Linux es compatible con plataformas amd64 (x86_64 / 64-Bit) e i386 (x86 / 32-Bit). Siempre que sea posible, el fabricante recomienda utilizar las imágenes amd64. Los requisitos de hardware son mínimos como se enumeran en la

⁷ <https://www.kali.org/downloads/>
⁸

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

⁹ <https://www.virtualbox.org/wiki/Downloads>



sección siguiente, aunque un mejor hardware naturalmente proporcionará un mejor rendimiento. Debería poder usar Kali Linux en hardware más nuevo con UEFI y sistemas más antiguos con BIOS.

Las imágenes i386, de forma predeterminada, utilizan un kernel PAE, por lo que puede ejecutarlas en sistemas con más de 4 GB de RAM.

En el ejemplo que se menciona más adelante, se instalará Kali Linux en una nueva máquina virtual invitada, sin ningún sistema operativo existente preinstalado.

Requisitos del sistema

Los requisitos de instalación para Kali Linux variarán según lo que le gustaría instalar y su configuración. Para conocer los requisitos del sistema:





En el extremo inferior, puede configurar Kali Linux como un servidor Secure Shell (SSH) básico sin escritorio, utilizando tan solo 128 MB de RAM (se recomiendan 512 MB) y 2 GB de espacio en disco.

En el extremo superior, si opta por instalar el escritorio Xfce4 predeterminado y el kali-linux-default metapaquete, realmente debería apuntar a al menos 2 GB de RAM y 20 GB de espacio en disco.

Cuando se utilizan aplicaciones que consumen muchos recursos, como Burp Suite, recomiendan al menos 8 GB de RAM (¡e incluso más si se trata de una aplicación web grande!) O utilizar programas simultáneos al mismo tiempo.

Requisitos previos de instalación¹⁰

Esta la guía se harán las siguientes suposiciones al instalar Kali Linux:

-  Usando la imagen del instalador de amd64.
-  Unidad de CD / DVD / soporte de arranque USB.
-  Disco único para instalar.
-  Conectado a una red (con DHCP y DNS habilitados) que tiene acceso a Internet saliente.

Preparación para la instalación




-  Descargue Kali Linux¹¹ (el fabricante recomienda¹² la imagen marcada como Instalador).

¹⁰ Dependiendo del tipo de instalación que seleccione, se pueden borrar todos los datos existentes en el disco duro, así que haga una copia de seguridad de la información importante del dispositivo en un medio externo.

¹¹ <https://www.kali.org/docs/introduction/download-official-kali-linux-images/>

¹² <https://www.kali.org/docs/introduction/what-image-to-download/#which-image-to-choose>



-  Grabe¹³ la ISO de Kali Linux en un DVD o una imagen de Kali Linux Live en una unidad USB. (Si no puede, consulte la instalación en red¹⁴ de Kali Linux).
-  Realice una copia de seguridad de la información importante del dispositivo en un medio externo.
-  Asegúrese de que su computadora esté configurada para arrancar desde CD / DVD / USB en su BIOS / UEFI.

Un vez que tiene preparado todos los materiales y el entorno para comenzar la instalación siga los pasos indicados en la sección “Kali Linux Installation Procedure” del siguiente enlace:

<https://www.kali.org/docs/installation/hard-disk-install/>



¹³ <https://www.kali.org/docs/usb/live-usb-install-with-windows/>

¹⁴ <https://www.kali.org/docs/installation/network-pxe/>



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 KALI¹⁵, pero si no fuere así puede instalarlos con los siguientes comandos, **previamente tomando privilegios de usuario “root”**:

```
# apt update && apt full-upgrade
(para verificar que su Sistema se encuentra actualizado con los parches)

#apt install tshark
(para instalar tshar)

# apt search ^tshark
Ordenando... Hecho
Buscar en todo el texto... Hecho
tshark/kali-rolling,now 3.4.4-1 amd64 [instalado, automático]
network traffic analyzer - console version
```

Nota: El símbolo “^” cumple la función de indicarle a la búsqueda que comience por el patrón indicado.

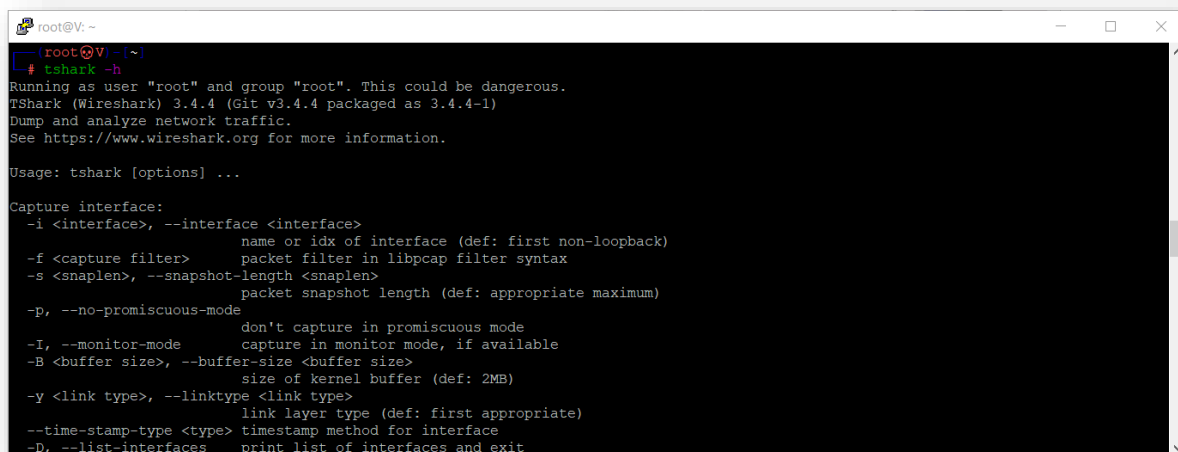
¹⁵ <https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/>



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”¹⁶.



```
root@V: ~  
[root@V] ~  
# tshark -h  
Running as user "root" and group "root". This could be dangerous.  
TShark (Wireshark) 3.4.4 (Git v3.4.4 packaged as 3.4.4-1)  
Dump and analyze network traffic.  
See https://www.wireshark.org for more information.  
  
Usage: tshark [options] ...  
  
Capture interface:  
-i <interface>, --interface <interface>  
                                name or idx of interface (def: first non-loopback)  
-f <capture filter>            packet filter in libpcap filter syntax  
-s <snaplen>, --snapshot-length <snaplen>  
                                packet snapshot length (def: appropriate maximum)  
-p, --no-promiscuous-mode      don't capture in promiscuous mode  
-I, --monitor-mode             capture in monitor mode, if available  
-B <buffer size>, --buffer-size <buffer size>  
                                size of kernel buffer (def: 2MB)  
-y <link type>, --linktype <link type>  
                                link layer type (def: first appropriate)  
--time-stamp-type <type>       timestamp method for interface  
-D, --list-interfaces          print list of interfaces and exit
```

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

```
# tshark -h  
Running as user "root" and group "root". This could be dangerous.  
TShark (Wireshark) 3.4.4 (Git v3.4.4 packaged as 3.4.4-1)  
Dump and analyze network traffic.  
See https://www.wireshark.org for more information.  
  
Usage: tshark [options] ...  
  
Capture interface:  
-i <interface>, --interface <interface>  
                                name or idx of interface (def: first non-loopback)  
-f <capture filter>            packet filter in libpcap filter syntax  
-s <snaplen>, --snapshot-length <snaplen>  
                                packet snapshot length (def: appropriate maximum)  
-p, --no-promiscuous-mode      don't capture in promiscuous mode  
-I, --monitor-mode             capture in monitor mode, if available  
-B <buffer size>, --buffer-size <buffer size>  
                                size of kernel buffer (def: 2MB)
```

¹⁶ La opción “-h” es relativamente estándar y cada comando debiera desplegar la ayuda de uso, en algunos casos deberá utilizar “--help”.



```
-y <link type>, --linktype <link type>
                                link layer type (def: first appropriate)
--time-stamp-type <type>        timestamp method for interface
-D, --list-interfaces           print list of interfaces and exit
-L, --list-data-link-types      print list of link-layer types of iface and exit
--list-time-stamp-types         print list of timestamp types for iface and exit

Capture stop conditions:
-c <packet count>               stop after n packets (def: infinite)
-a <autostop cond.> ..., --autostop <autostop cond.> ...
                                duration:NUM - stop after NUM seconds
                                filesize:NUM - stop this file after NUM KB
                                files:NUM - stop after NUM files
                                packets:NUM - stop after NUM packets

Capture output:
-b <ringbuffer opt.> ..., --ring-buffer <ringbuffer opt.>
                                duration:NUM - switch to next file after NUM secs
                                filesize:NUM - switch to next file after NUM KB
                                files:NUM - ringbuffer: replace after NUM files
                                packets:NUM - switch to next file after NUM packets
                                interval:NUM - switch to next file when the time is
                                                an exact multiple of NUM secs

Input file:
-r <infile>, --read-file <infile>
                                set the filename to read from (or '-' for stdin)

Processing:
-2                               perform a two-pass analysis
-M <packet count>               perform session auto reset
-R <read filter>, --read-filter <read filter>
                                packet Read filter in Wireshark display filter syntax
                                (requires -2)
-Y <display filter>, --display-filter <display filter>
                                packet display filter in Wireshark display filter
                                syntax
-n                               disable all name resolutions (def: all enabled)
-N <name resolve flags>         enable specific name resolution(s): "mnNtdv"
-d <layer_type>==<selector>,<decode_as_protocol> ...
                                "Decode As", see the man page for details
                                Example: tcp.port==8888,http
-H <hosts file>                 read a list of entries from a hosts file, which will
                                then be written to a capture file. (Implies -W n)
--enable-protocol <proto_name> enable dissection of proto_name
--disable-protocol <proto_name> disable dissection of proto_name
--enable-heuristic <short_name> enable dissection of heuristic protocol
--disable-heuristic <short_name> disable dissection of heuristic protocol

Output:
-w <outfile|->                  write packets to a pcapng-format file named "outfile"
                                (or '-' for stdout)
--capture-comment <comment>    set the capture file comment, if supported
-C <config profile>             start with specified configuration profile
-F <output file type>           set the output file type, default is pcapng
                                an empty "-F" option will list the file types
```



-V	add output of packet tree (Packet Details)
-O <protocols>	Only show packet details of these protocols, comma separated
-P, --print	print packet summary even when writing to a file
-S <separator>	the line separator to print between packets
-x	add output of hex and ASCII dump (Packet Bytes)
-T pdml ps psml json jsonraw ek tabs text fields ?	format of text output (def: text)
-j <protocolfilter>	protocols layers filter if -T ek pdml json selected (e.g. "ip ip.flags text", filter does not expand child nodes, unless child is specified also in the filter)
-J <protocolfilter>	top level protocol filter if -T ek pdml json selected (e.g. "http tcp", filter which expands all child nodes)
-e <field>	field to print if -Tfields selected (e.g. tcp.port, _ws.col.Info)
-E<fieldsoption>=<value>	this option can be repeated to print multiple fields
bom=y n	set options for output when -Tfields selected:
header=y n	print a UTF-8 BOM
separator=/t /s <char>	switch headers on and off
occurrence=f l a	select tab, space, printable character as separator
aggregator=,/s <char>	print first, last or all occurrences of each field
quote=d s n	select comma, space, printable character as aggregator
-t a ad adoy d dd e r u ud udoy	select double, single, no quotes for values
-u s hms	output format of time stamps (def: r: rel. to first)
-l	output format of seconds (def: s: seconds)
-q	flush standard output after each packet
-Q	be more quiet on stdout (e.g. when using statistics)
-g	only log true errors to stderr (quieter than -q)
-W n	enable group read access on the output file(s)
-X <key>:<value>	Save extra information in the file, if supported.
-U tap_name	n = write network address resolution information
-z <statistics>	eXtension options, see the man page for details
--export-objects <protocol>,<destdir>	PDU's export mode, see the man page for details
	various statistics, see the man page for details
--color	save exported objects for a protocol to a directory named "destdir"
	color output text similarly to the Wireshark GUI, requires a terminal with 24-bit color support
	Also supplies color attributes to pdml and psml
formats	
--no-duplicate-keys	(Note that attributes are nonstandard)
object	If -T json is specified, merge duplicate keys in an object
all	into a single key with as value a json array containing values
--elastic-mapping-filter <protocols>	If -G elastic-mapping is specified, put only the specified protocols within the mapping file
Miscellaneous:	
-h, --help	display this help and exit
-v, --version	display version info and exit
-o <name>:<value> ...	override preference setting
-K <keytab>	keytab file to use for kerberos decryption



```
-G [report]          dump one of several available reports and exit  
                     default report="fields"  
                     use "-G help" for more help
```

Dumpcap can benefit from an enabled BPF JIT compiler if available.
You might want to enable it by executing:
"echo 1 > /proc/sys/net/core/bpf_jit_enable"
Note that this can make your system less secure!



Paso 4: Ponerlo en marcha para verificar nuestra infraestructura.

Un ejemplo de ejecución básica para nuestros primeros pasos:

Probaremos el comando TSHARK con nuestro KALI y bajo la modalidad de observación del tráfico de la interfaz de red eth0 (depende de cada computador el nombre de la interfaz de red). Recuerde que son importantes comandos de Linux como “ifconfig”, “netstat” y archivos como “/etc/services”.

EJEMPLO 1 - TSHARK OBSERVANDO EL TRÁFICO HTTP (80/TCP)

```
# tshark -f "tcp port 80" -i eth0

-f <filtro de captura> filtro de paquetes en la sintaxis del filtro libpcap
-i <interface> nombre o idx de la interfaz (def: primer no loopback)
```

Donde podemos observar algo como lo siguiente en la consola:

```
root@V: ~
(root@V) ~
# tshark -f "tcp port 80" -i eth0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
  1 0.000000000 45.4.2.67 → 181.212.41.74 TCP 66 1030 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
  2 0.000795143 45.4.2.67 → 181.212.41.74 TCP 66 14178 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
  3 0.262150625 45.4.2.67 → 181.212.41.74 TCP 66 1039 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
  4 1.009706801 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1030 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
  5 1.009706974 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 14178 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
  6 1.260653467 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1039 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
  7 3.016108305 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1030 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
  8 3.016909503 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 14178 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
  9 3.268982992 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1039 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
 10 7.024890303 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1030 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
 11 7.025032648 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 14178 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_P
ERM=1
```

EJEMPLO 2 -TSHARK OBSERVANDO EN TRÁFICO DNS (53/UDP)

```
# tshark -f "udp port 53" -i eth0

-f <filtro de captura> filtro de paquetes en la sintaxis del filtro libpcap
-i <interface> nombre o idx de la interfaz (def: primer no loopback)
```




Donde podemos observar algo como lo siguiente en la consola:

```
root@V: ~
[~]
# tshark -f "udp port 53" -i eth0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
1 0.000000000 181.212.41.74 → 8.8.8.8      DNS 73 Standard query 0xcdf A www.google.cl
2 0.000010542 181.212.41.74 → 8.8.8.8      DNS 73 Standard query 0xf325 AAAA www.google.cl
3 0.001712934   8.8.8.8 → 181.212.41.74  DNS 89 Standard query response 0xcdf A www.google.cl A 64.233.186.94
4 0.001850373   8.8.8.8 → 181.212.41.74  DNS 101 Standard query response 0xf325 AAAA www.google.cl AAAA 2800:3f0:4003:c02::5e
5 0.003908430 181.212.41.74 → 8.8.8.8      DNS 86 Standard query 0xf054 PTR 94.186.233.64.in-addr.arpa
6 0.005663133   8.8.8.8 → 181.212.41.74  DNS 119 Standard query response 0xf054 PTR 94.186.233.64.in-addr.arpa PTR cb-in-f94.1e10
0.net
7 1.005563357 181.212.41.74 → 8.8.8.8      DNS 86 Standard query 0xc82d PTR 94.186.233.64.in-addr.arpa
8 1.007028711   8.8.8.8 → 181.212.41.74  DNS 119 Standard query response 0xc82d PTR 94.186.233.64.in-addr.arpa PTR cb-in-f94.1e10
0.net
9 2.006953791 181.212.41.74 → 8.8.8.8      DNS 86 Standard query 0xa987 PTR 94.186.233.64.in-addr.arpa
10 2.009164684   8.8.8.8 → 181.212.41.74  DNS 119 Standard query response 0xa987 PTR 94.186.233.64.in-addr.arpa PTR cb-in-f94.1e10
0.net
11 3.007996183 181.212.41.74 → 8.8.8.8      DNS 86 Standard query 0xc377 PTR 94.186.233.64.in-addr.arpa
12 3.009617074   8.8.8.8 → 181.212.41.74  DNS 119 Standard query response 0xc377 PTR 94.186.233.64.in-addr.arpa PTR cb-in-f94.1e10
0.net
13 4.009514264 181.212.41.74 → 8.8.8.8      DNS 86 Standard query 0xc9b5 PTR 94.186.233.64.in-addr.arpa
14 4.011395188   8.8.8.8 → 181.212.41.74  DNS 119 Standard query response 0xc9b5 PTR 94.186.233.64.in-addr.arpa PTR cb-in-f94.1e10
0.net
```

EJEMPLO 2 -TSHARK OBSERVANDO EN TRÁFICO HTTPS (443/TCP)

```
# tshark -f "tcp port 443" -i eth0
```

-f <filtro de captura> filtro de paquetes en la sintaxis del filtro libpcap
-i <interface> nombre o idx de la interfaz (def: primer no loopback)

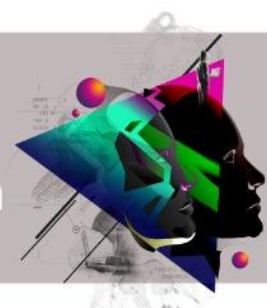
Donde podemos observar algo como lo siguiente en la consola:

```
root@V: ~
[~]
# tshark -f "tcp port 443" -i eth0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
1 0.000000000 45.4.2.67 → 181.212.41.74 TCP 66 12578 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
2 0.000101302 45.4.2.67 → 181.212.41.74 TCP 66 1026 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
3 0.266676712 45.4.2.67 → 181.212.41.74 TCP 66 23410 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
4 1.011132850 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 12578 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
5 1.011133022 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1026 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
6 1.277572609 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 23410 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
7 3.018024746 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 12578 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
8 3.018024919 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1026 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
9 3.286874697 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 23410 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
10 7.025335701 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 12578 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
11 7.025335858 45.4.2.67 → 181.212.41.74 TCP 66 [TCP Retransmission] 1026 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1440 WS=256 SACK_PERM=1
```

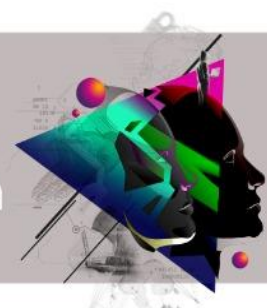


Otros ejemplos de uso para aplicar en sus respectivos entornos:

Objetivo de TSHARK	Como se aplica al comando TSHARK
Mostrar las interfaces disponibles	tshark -D
Mostrar la ayuda	tshark -h
Capturar el tráfico de una interfaz	tshark -i # (donde # es el número de interfaz obtenido desde el comando de arriba -D)
	tshark -i 'name' (donde 'name' es el nombre de interfaz obtenido desde el comando de arriba -D)
Escribir los paquetes capturados a un archivo	tshark -i # -w {ruta de directorio y nombre de archive: /tmp/miarchivo.pcapng por ejemplo}
Capturar usando filtros	tshark -i # -f "filter text using BPF syntax"
	ejemplo: tshark -i 5 -f "tcp port 80"
Leer una captura genérica para un dirección IP específica de un archivo capture.pcapng	tshark -R "ip.addr == 192.168.0.1" -r /tmp/capture.pcapng
Ethernet address 00:08:15:00:08:15	eth.addr == 00:08:15:00:08:15
Ethernet type 0x0806 (ARP)	eth.type == 0x0806
Ethernet broadcast	eth.addr == ff:ff:ff:ff:ff:ff
No ARP	not arp
Mostrar solo tráfico IPv4	ip
Mostrar solo tráfico IPv6	ip6
Mostrar todo IPv4 menos lo que incluya a 192.168.0.1, no use != para esto!	!(ip.addr == 192.168.0.1)
Mostrar solo el tráfico IPX	ipx
Mostrar solo el tráfico TCP	tcp
Mostrar solo el tráfico UDP	udp



Incluir filtros de despliegue en el comando cuando se analiza un archivo capturado	-Y <display filter>
Tráfico UDP que no sea del puerto 53 (no DNS), no use != for this!	tshark -i 1 -Y "!(udp.port == 53)"
TCP o UDP con puerto 80 (HTTP)	tcp.port == 80 udp.port == 80
Solamente HTTP	http
No ARP y no DNS	not arp and not (udp.port == 53)
No HTTP y no SMTP hacia y desde la IP 192.168.0.1	not (tcp.port == 80) and not (tcp.port == 25) and ip.addr == 192.168.0.1
Crear un archivo con separador “;” con los campos “source IP” “destination IP” and “Destination Port” a partir de todas las conexiones iniciadas con SYN. Use las opciones -T, -E y -e.	tshark -nn -r capturefile.dmp -T fields -E separator=';' -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport '(tcp.flags.syn == 1 and tcp.flags.ack == 0)'
Despliegue los códigos de respuesta http	tshark -o "tcp.desegment_tcp_streams:TRUE" -i eth0 -R "http.response" -T fields -e http.response.code
Despliegue las Top 10 URLs	tshark -r capture.pcapng -R http.request -T fields -e http.host -e http.request.uri
	sed -e 's/?.*\$/ /' sed -e 's#^(.*)t(.*)\$#http://12#' sort uniq -c sort -rn head
Despliegue Source IP y la dirección MAC Address. (con “;” como separador)	tshark -i eth0 -nn -e ip.src -e eth.src -Tfields -E separator=, -R ip
Despliegue la IP objetivo and la dirección Mac (con “;” como separador)	tshark -i eth0 -nn -e ip.dst -e eth.dst -Tfields -E separator=, -R ip
Source and Target IPv4	tshark -i eth0 -nn -e ip.src -e ip.dst -Tfields -E separator=, -R ip



Source and Target IPv6	tshark -i eth0 -nn -e ip6.src -e ip6.dst -Tfields -E separator=, -R ip6
Consultas DNS por IP de origen	tshark -i eth0 -nn -e ip.src -e dns.qry.name -E separator=";" -T fields port 53
Desplegar solamente la IP de origen y de destino	tshark -o column.format:"Source", "%s", "Destination", "%d" -Ttext
Estadísticas desde un archive capturado y almacenandolas en stat.txt:	tshark -r capture.pcapng -qz io,stat,1,0,sum(tcp.analysis.retransmission)"ip.addr==10.10.10.10" > stat.txt
Pruebe estos comandos para generar diferentes estadísticas	tshark -r capture.pcapng -qz io,stat,120,"ip.addr==194.134.109.48 && tcp","COUNT(tcp.analysis.retransmission)ip.addr==194.134.109.48 && tcp.analysis.retransmission"
En el siguiente ejemplo se usa el nombre de archive capture.pcapng, solo sustituyalo por el nombre de su archive de captura para los analisis de su preferencia.	tshark -r samples.cap -q -z io,stat,30,"COUNT(tcp.analysis.retransmission) tcp.analysis.retransmission"
	tshark -r capture.pcapng -q -z io,stat,30,
	"COUNT(tcp.analysis.retranmission)tcp.analysis.retransmission",
	"AVG(tcp.window_size)tcp.window_size,tAЭ,tAЭMAX(tcp.window_size)",
	"MIN(tcp.window_size)tcp.window_size"
	tshark -r capture.pcapng -q -z io,stat,5,"COUNT(tcp.analysis.retransmission) tcp.analysis.retransmission","COUNT(tcp.analysis.duplicate_ack)tcp.analysis.duplicate_ack",
	"COUNT(tcp.analysis.lost_segment) tcp.analysis.lost_segment",
	"COUNT(tcp.analysis.fast_retransmission) tcp.analysis.fast_retransmission"
	tshark -r capture.pcapng -q -z io,stat,5,"MIN(tcp.analysis.ack_rtt)tcp.analysis.ack_rtt",
	"MAX(tcp.analysis.ack_rtt)tcp.analysis.ack_rtt","AVG(tcp.analysis.ack_rtt)t cp.analysis.ack_rtt"
	tshark -r capture.pcapng -q -z ip_hosts,tree
	tshark -r capture.pcapng -q -z conv,tcp
	tshark -r capture.pcapng -q -z ptype,tree



Tenga presente que es importante que estas pruebas sean coordinadas con el equipo de operaciones y en ambientes que estén bajo supervisión.

Antes de proceder a aplicar estos comandos revise sus políticas de seguridad de la información interna, sus códigos de ética, los NDA que haya suscrito y las cláusulas de confidencialidad de su contrato de trabajo.

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.

Tenga presente que para el procesamiento y análisis de los datos es relevante que vaya perfeccionando su manejo de LINUX y comandos PowerShell (si es un usuario de windows).

En próximas ediciones se irán reforzando estos aspectos para facilitar el manejo de los datos y resultados obtenidos, logrando así una mejor comunicación con sus equipos TIC y con el CSIRT de Gobierno.

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



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

Comandos Linux/Unix de manipulación de archivos y directorios:

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



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.