Exercicis de comandes de Linux relacionades amb el Nivell 7 del model TCP/IP (HTTP)

1.-Llegeix primer els apartats "HTTP Flow" i "HTTP messages" que hi ha al final de l'article https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview per recordar com funciona el protocol HTTP i saber què són els mètodes, els codis de resposta, les capçaleres de client i les capçaleres de servidor. A partir d'aquí...:

> NOTA: A l'article no s'indica, però recordeu que la capçalera de client "Host:" sempre és obligatòria. Serveix per indicar el nom DNS del servidor web al què es vol accedir, ja que una mateixa màquina (IP) podria tenir diferents llocs webs i és per això que s'ha de dir a quin es vol accedir

a) ...arrenca una màquina virtual qualsevol i utilitza la comanda *ncat* (si no està instal.lada, instal.la-la) com a client HTTP per tal d'obtenir la pàgina http://www.redhat.com/en/open-source Mira la pista de sota per saber com fer-ho. ¿Quin codi de resposta i codi HTML veus? Per què? Fixa't en el valor de la capçalera de resposta anomenada "Location".

> PISTA: Executa la comanda així: *ncat www.redhat.com 80* i, escriu, al terminal que apareix:
> *GET /en/open-source HTTP/1.1*
> *Host:www.redhat.com*
> *(línia en blanc)*
>
> NOTA: Si en comptes de fer la petició interactivament la volguessir fer mitjançant la comanda *echo* "entubant-la" al *ncat* directament, recorda d'afegir llavors a l'*echo* el paràmetre -e per a què els símbols \r\n s'interpretin correctament com a salts de línia, així: *echo -ne "GET /en/open-source HTTP/1.1\r\nHost:www.redhat.com\r\n\r\n" | ncat www.redhat.com 80*

b) Utilitza el mateix *ncat* pero ara com a client HTTPS (és a dir, HTTP + TLS) per tal d'obtenir la pàgina https://www.redhat.com/en/open-source Mira la pista de sota per saber com fer-ho. ¿Quin codi de resposta i codi HTML veus ara? Per què?

> PISTA: L'únic canvi que has de fer respecte l'apartat anterior és afegir el paràmetre *--ssl* a la comanda *ncat* i canviar el port de destí per a què sigui 443 en comptes de 80

c) Utilitza el mateix *ncat* pero ara com a servidor HTTP. Concretament, primer crea un fitxer anomenat "pag.html" amb el següent contingut...:

```
<html><body><h1>Benvingut</h1><p>M'agraden els macarrons</p></body></html>
```

...i després crea un script anomenat "servWeb.sh" amb el següent contingut:

```
#!/bin/bash
while [[ true ]] ; do
        { echo -ne "HTTP/1.1 200 OK\r\nContent-Length: $(wc -c pag.html)\r\n\r\n"; cat pag.html; } | ncat -l -p 8080
done
```

> NOTA: La capçalera "Content-Length" és necessària enviar-la per dir-li al client el tamany de la pàgina web oferida i així fer-li saber quan l'ha rebut tota (i per tant quan no cal continuar tenint oberta la conexió): si no es digués aquest tamany el client es quedaria esperant a rebre més dades fins que finalment arribés el temps de tallar la conexió (timeout), però aquest comportament no és gens òptim.
> NOTA: Les claus { } són importants perquè fan que la canonada rebi la sortida no només de la comanda *cat* sinó també de la comanda *echo* prèvia (i en el mateix ordre)

Executa l'script anterior i seguidament obre un navegador per escriure la direcció http://127.0.0.1:8080 a la seva barra de direccions. ¿Què veus? ¿Què apareix al terminal on s'està executant l'script? ¿Per què creus que no s'ha fet servir a l'script el paràmetre *-k* de *ncat*?

> NOTA: Si al navegador obtens un error de conexió és possible que tinguis un tallafocs activat al teu sistema. Deshabilita'l temporalment amb la comanda *sudo systemctl stop firewalld* o *systemctl stop ufw* i torna-ho a intentar.
> NOTA: Si es volgués fer servir *ncat* com a servidor HTTPS, a més dels paràmetres *-l* i *-p* caldria indicar també el paràmetre *--ssl* ja conegut però, a més, dos paràmetres extra més: *--ssl-cert (*per indicar la ruta del certificat a usar) i *--ssl-key* (per indicar la ruta de la clau privada a usar); el tema dels certificats+claus privades ho estudiarem més endavant.

2.-a) ¿A quin sol paràmetre de *curl* és equivalent escriure el conjunt dels paràmetres *-s, -D -* i *-o /dev/null* (així per exemple: *curl -s -D - -o /dev/null https://www.hola.com*)? Consulta la referència de teoria si cal.

b) Explica per què mostren el que mostren les següents comandes (la pots provar al terminal de la teva màquina real o al d'un màquina virtual qualsevol si instal.les abans els paquet "curl"):

curl -sIL https://www.twitter.com | grep "HTTP/"
Raona per què aquesta comanda es podria fer servir per saber si un determinat lloc web està caigut

curl -sIL https://buff.ly/1lTcZSM | grep "^Location"
NOTA: Aquí la clau està en el valor de la capçalera de servidor Location

curl -s http://tinyurl.com/api-create.php?url=http://www.google.com

curl -s https://www.merriam-webster.com/word-of-the-day | grep -Eo "word-of-the-day/.-$(date +%Y-%m-%d)" | uniq | cut -f2 -d"/" | cut -f1 -d"-"*
NOTA: Fixa't sobre tot en l'expressió regular utilitzada al *grep*. Pots consultar com funcionen les expressions regulars a http://www.regular-expressions.info i/o provar-les a http://regexr.com o https://regex101.com

curl -s http://artscene.textfiles.com/vt100/wineglas.vt | pv -L4800 -q
NOTA: El contingut dels arxius "vt" (hi ha molts més a http://artscene.textfiles.com/vt100) són seqüències de control del terminal per posicionar el cursor a la pantalla, fer-lo desaparèixer, etc; es pot trobar una llista completa a http://www.termsys.demon.co.uk/vtansi.htm . La comanda *pv* serveix per mostrar el progrés de les dades que rep per la canonada (encara que amb *-q* no mostra res de fet, perquè el que realment es vol és fer servir la funcionalitat del seu paràmetre *-L*, el qual serveix per modificar la ràtio de transferència de dades en bytes/s (per tant, si l'animació va massa ràpida o massa lenta, es pot canviar aquest valor).

c) Prova aquestes URL curioses al terminal i digues què es veu:

curl https://wttr.in/Barcelona

curl -A "Mozilla/5.0 (Windows NT 10.0; Win64; x64)" https://wttr.in/Barcelona

curl http://artscene.textfiles.com/asciiart/panda

curl -T gat.jpg https://free.keep.sh
Consulta el manual de *curl* per saber per a què serveix el seu paràmetre *-T* i la documentació del servei "Keep.sh" (https://keep.sh/docs) per saber què ofereix, com funciona i altres exemples d'ús.
NOTA: Una altra URL que serveix pel mateix és http://filepush.co/upload (es faria servir així: *curl -T gat.jpg http://filepush.co/upload/gat.jpg*). Altres similar són https://transfersh.com o https://transfer.sh

curl https://ipinfo.io
NOTA: Més URLs que mostren la teva IP pública (i més dades relacionades) en format JSON de forma similar a l'anterior són: http://httpbin.org/ip o http://wtfismyip.com/json Més URLs que mostren (només) la teva IP pública (sense salt de línia al final) són http://whatismyip.akamai.com, http://ifconfig.me , http://l2.io/ip , http://ipecho.net/plain , http://ident.me , http://canihazip.com/s , http://tnx.nl/ip , http://wgetip.com, http://ip.tyk.nu, http://bot.whatismyipaddress.com, http://curlmyip.net Més URLs que mostren (només) la teva IP pública (amb salt de línia al final) són http://icanhazip.com, http://eth0.me, http://ipaddr.site, http://ifconfig.co, http://ifconfig.pro, http://ipinfo.io/ip, http://checkip.amazonaws.com .També pots provar amb *curl* aquests serveis indicant altres direccions IP pública que no siguin la teva per obtenir-ne informació relacionada (geolocalització, operador telefònic, etc. Per exemple: http://ipinfo.io/207.46.13.41 (o http://ipinfo.io/207.46.13.41/loc) , http://ifconfig.me/207.46.13.41, http://ifconfig.co/country (o http://ifconfig.co/city , http://ifconfig.co/country-iso , http://ifconfig.co/json)

d) Curl no només és un client HTTP/S: també és client de molts altres protocols més "obscurs" com ara DICT (https://en.wikipedia.org/wiki/DICT). Sabent això, ¿què creus que fa aquesta comanda: *curl dict://dict.org/d:unix*

e) Crea un "toilet" a https://www.ptsv2.com i a continuació envia una petició POST a https://www.ptsv2.com/t/*nom_del_toilet*/post amb les següents dades (estan en format JSON, compte amb indicar el valor correcte a la capçalera de client "Content-Type"): *{"nom":"ana","edat":26}* . Comprova finalment que aquestes dades hagin arribat bé anant a la pàgina *https://www.ptsv2.com/t/nom_del_toilet*

PISTA: Per realitzar una petició POST (com és el cas) en comptes del tipus per defecte, que és GET, cal escriure al *curl* el paràmetre ***-X POST.*** No obstant, aquest paràmetre ha de venir acompanyat sempre d'un altre on s'especifiqui exactament les dades concretes a enviar al servidor: aquest paràmetre té la forma general *-d "dades"*, així que en aquest cas, doncs, haurem d'afegir això també: ***-d '{"nom":"ana","edat":26}'*** No obstant, el format per defecte de les dades que s'envien en una petició POST estàndard està estipulat que sigui "application/x-www-form-urlencoded" , així que per indicar al servidor que a la petició POST s'envien dades en format JSON cal afegir un altre paràmetre que serveix per informar d'aquest fet, concretament ***-H 'Content-Type:application/json'*** (aquest paràmetre el que fa en realitat simplement és afegir a la petició la capçalera de client indicada, la qual pot ser qualsevol però en aquest cas concret és "Content-Type" , la qual serveix justament per informar del tipus MIME de les dades que acompanya a la petició

NOTA: Tal com hem dit, si no modifiquéssim explícitament el valor per defecte de la capçalera de petició "Content-Type" el format de les dades enviades via POST seria "application/x-www-from-urlencoded". Això implica que el paràmetre *-d* hauria d'indicar-se llavors així: ***-d '?nom=ana&edat=26'*** En qualsevol cas, es faci servir un format o un altre, si les dades estiguessin guardades en un fitxer, llavors hauríem d'utilitzar el paràmetre *-d* així: ***-d @/ruta/fitxer.txt***

3.-a) ¿Què fa la línia *curl* ***-O*** *-C -* *http://www.linuxvoice.com/issues/016/Linux-Voice-Issue-016.pdf*? ¿Quina diferència hi ha amb *curl* ***-o magazine16.pdf*** *-C - http://www.linuxvoice.com/issues/016/Linux-Voice-Issue-016.pdf* ? ¿I amb *curl* ***-o /opt/magazine16.pdf*** *-C - http://www.linuxvoice.com/issues/016/Linux-Voice-Issue-016.pdf ?*

b) Per descarregar tots els Pdfs que hi ha penjats a la web anterior (http://www.linuxvoice.com/issues), gràcies a què les seves URLs segueixen una estructura amb un patró clar i definit, és fàcil realitzar un script que vagi descarregant-los un a un, com el següent. Prova'l i raona el per què de la condició que hi apareix:

```
#/bin/bash
for i in {1..50}
do
    if [[ $i -lt 10 ]] ;  then
        curl -O -C - http://www.linuxvoice.com/issues/00$i/Linux-Voice-Issue-00$i.pdf
    else
        curl -O -C - http://www.linuxvoice.com/issues/0$i/Linux-Voice-Issue-0$i.pdf
    fi
done
```

NOTA: Els enllaços a la web de LinuxVoice també podrien ser de tipus "https" en comptes de "http" però llavors caldria afegir el paràmetre *-k* a la comanda *curl* per a què aquesta els acceptés. Això és degut a què aquesta web no implementa correctament la seguretat proporcionada per "https" (o, en altres paraules, ofereix un certificat "de mala qualitat" perquè és autosignat) i per defecte curl no accepta aquest tipus d'enllaços (a no ser que se l'indiqui el contrari amb el mencionat paràmetre *-k*)
NOTA: En realitat, *curl* admet la possibilitat d'indicar sèries senzilles a l'hora de descarregar varis fitxers amb una sola comanda si fem servir els claudàtors i/o les claus, com mostren els següents exemples hipotètics (on el paràmetre -Z serveix per fer múltiples descàrregues en paral·lel en lloc de seqüencialment). De fet, aquest apartat es podria fer d'aquesta manera:

```
curl -Z -O "https://example.com/section[A-Z].html"      : Descarrega "sectionA.html", "sectionB.html", etc
curl -Z -O "https://example.com/index-[01-10].html"     : Descarrega "index01.html", "index02.html", etc
curl -Z -O "https://example.com/index-[01-10:2].html"   : Descarrega "index01.html", "index03.html", etc
curl -Z -O "https://example.com/{one,beta,plus}.html"   : Descarrega "one.html", "beta.html" i "plus.html"
```

c) ¿Què fa la línia *curl -s https://elpais.com | grep -Eo '[^"]*\.(png|jpg)'* ? A partir d'aquí, ¿què faria el següent script?

```
#!/bin/bash
for foto in $(curl -s https://elpais.com | grep -Eo '[^"]*\.(png|jpg)' ) ;  do
        curl -O "$foto"
done
```

NOTA: Per realitzar descàrregues massives de fitxers vía HTTP/S és més còmode, no obstant, utilitzar la comanda *wget*, més especialitzada en aquest tipus de tasques (tal com veurem al proper exercici)

d) Què fa aquest script? Estudia tots els detalls (el que retorna *curl*, els paràmetres del *grep*, la sintaxi *${#nomArray[@]}*, etc)

```
#!/bin/bash
ipPublica=$(curl -s ipecho.net/plain)
#Els parèntesis exteriors són per generar un array amb els valors obtinguts de $(nmap ... | grep ...)
arrayPorts=($(nmap ${ipPublica} | grep -Eo "^[0-9]{1,5}"))
```

```
for (( i = 0; i < ${#arrayPorts[@]}; i++ )) ;  do
        echo -e "PORT: ${arrayPorts[$i]}"
done
```

e) I aquest? (fixa't en el paràmetre *-w* de *curl*; consulta el manual si ho necessites)

```
#!/bin/bash
for i in {1..3} ;  do
        curl -s -w "Temps de càrrega: %{time_total}\n" -o /dev/null https://www.google.com
done
```

NOTA: Altres variables interessants que es poden indicar al missatge indicat amb *-w* són, per exemple: *%{response_code}, %{content_type}, %{num_redirects}, %{url_effective}, %{remote_ip}, %{remote_port}, %{size_download}, %{speed_download}, %{time_connect}, %{time_appconnect}, %{time_namelookup},...* Per més informació, consultar el manual de *curl* o també https://blog.kenweiner.com/2014/11/http-request-timings-with-curl.html

4.-a) Obre el Firefox de la màquina real i vés a una pàgina web qualsevol ¿Què veus si pulses llavors CTRL+SHIFT+E i tot seguit F5 (per repetir la petició)? Digues el significat de les següents columnes que apareixen: "Estat", "Mètode", "Domini", "Fitxer", "Tipus", "Mida"

NOTA: Fixa't que a la zona superior dreta tens la possibilitat de filtrar la llista mostrada de peticions segons el tipus de contingut demanat (HTML, CSS, Javascript, imatges, etc). I també la possibilitat de deshabilitar la catxé del navegador (per tal de realitzar les peticions sempre al destí real)

aII) Clica amb el botó dret sobre el títol de qualsevol de les columnes mostrades i sel·lecciona les opcions "IP remota", "Protocol", "Galetes" i "Temps"->"Durada". ¿Què veus en cada cas?

b) Si sel.lecciones una petició qualsevol de les que t'hagin aparegut, digues el significat dels següents apartats que apareixen sota la pestanya "Capçaleres" dins de la zona dreta de la pantalla: "Capçaleres de resposta", "Capçaleres de sol.licitud", botó "Edita i torna a enviar"" ¿I el de la informació que apareix sota la pestanya "Temps"? ¿I la pestanya "Memoria cau", què representa que és? ¿I "Seguretat"?

NOTA: Aquesta zona dreta informa de moltes coses més (pestanya "Galetes", "Paràmetres", "Resposta", "Temps", "Seguretat") però encara és massa aviat per estudiar-les. Més endavant ho farem

c) Sel.lecciona una petició que sigui de tipus imatge ("png", "jpg", etc) i fes-hi clic amb el botó dret del ratolí. Al menú contextual que apareix, per a què serviria l'opció "Anomena i desa la imatge"? ¿I l'opció "Copia->Copiar com a curl"?

La comanda de terminal *wget* és un descarregador de fitxers a través dels protocols HTTP/S, FTP/S i més. És una eina més especialitzada que *curl*: mentre que aquest és un client HTTP/S genèric que, entre altres coses, permet descarregar fitxers, *wget* és un client HTTP/S específicament dissenyat per aquesta tasca concreta: descarregar (i, per tant, ofereix més opcions específiques que *curl* no dóna, com ara la possibilitat de fer descàrregues recursives, entre d'altres). Tot seguit es mostra una "xuleta" de les opcions més útils:

wget https://url/dun/fitxer	Descarrega al disc dur el fitxer indicat
-c	Continua la descàrrega (si anteriorment va fallar) des d'allà on es va interrompre
-O nom	Indica el nom que tindrà el fitxer un cop descarregat
-r	Realitza una descàrrega recursiva si la URL indicada és la d'una carpeta en comptes de la d'un fitxer. Combinat amb el paràmetre *-l nº* serveix per indicar fins a quin nivell (1=una subcarpeta, 2= dues subcarpetes) es vol descarregar...si no s'indica s'entén "infinit"
-p	Realitza la descàrrega de tots els recursos (CSS, Javascript, imatges...) referenciades per la pàgina descarregada, si la URL indicada és la d'una pàgina individual, per tal de què aquesta es pugui visualitzar adientment. Se sol combinar amb *-H* per tal de permetre connectar a altres servidors més enllà de l'indicat per si cal descarregar-hi de més enllà aquests recursos
-N	Descarrega només els arxius més nous que els locals
-A "ext1","ext2",...	Descarrega només els arxius que trobi amb l'extensió indicada El paràmetre contrari (descarrega tot menys els arxius indicats) és *-R*
-np	No descarrega contingut anterior a la URL indicada. Per a què funcioni correctament aquest paràmetre, caldria que la URL indicada finalitzés amb el símbol "/"
-nd	Tot ho descarrega a la mateixa carpeta local (sense respectar, doncs, la jerarquia de carpetes del lloc remot)
-k	Un cop feta la descàrrega, transforma els enllaços per tal de què tot el contingut es pugui visitar offline (canvia les rutes absolutes per relatives i als recursos no descarregats els referencia amb la URL completa). Se sol combinar amb *-E*, el qual afegeix l'extensió adient a cada recurs descarregat (".css" als arxius CSS, ".js" als arxius Javascript, etc) ja que per defecte *wget* això no ho fa i això pot provocar que els links (que sí incorporen l'extensió) no apuntin al nom correcte del recurs.
-U valorUserAgent	Permet indicar el valor desitjat per la capçalera e petició "User-Agent". Aquest paràmetre és útil quan el lloc a accedir té prohibit l'entrada específicament a *wget* (i cal llavors enganyar-lo fent-li creure que som un navegador normal)
-e robots=off	Permet saltar-se la restricció de fer cas a l'arxiu "robots.txt". Aquest paràmetre és útil per poder accedir a totes les subcarpetes on es troben els recursos a descarregar, ja que pot ser que existeixi al servidor en qüestió un arxiu "robots.txt" que no ho permeti (consulta https://www.robotstxt.org/robotstxt.html per + informació)

5.- Consulta la llista de paràmetres de la comanda *wget* indicada al paràgraf anterior i digues quina combinació d'aquests utilitzaries per descarregar-te en una mateixa carpeta de la teva màquina real tots (i només) els arxius PDF que penjen recursivament sota la URL https://www.linuxvoice.com/issues Prova-ho