

# Exercici0

Investiga que fa aquesta comanda `python -m http.server 8000` i comenta el que fa.

Fes que el teu PC (pots fer-lo a windows o a Linux) sigui un servidor web python i que el teu company es pugui descarregar un fitxer que es digui `EITeuNomCognomExercici0.txt` que hagi creat al teu directori personal mitjançant aquest servidor.

**NOTA:** no oblidis que a l'instal·lar python també hem afegir-lo al path del teu Sistema Operatiu, per poder executar python des de qualsevol directori.

Has de mostrar l'execució de la comanda python al teu servidor i la Ip del servidor. També el navegador web del teu company connectant-se a la teva IP on aparegui el fitxer a descarregar.

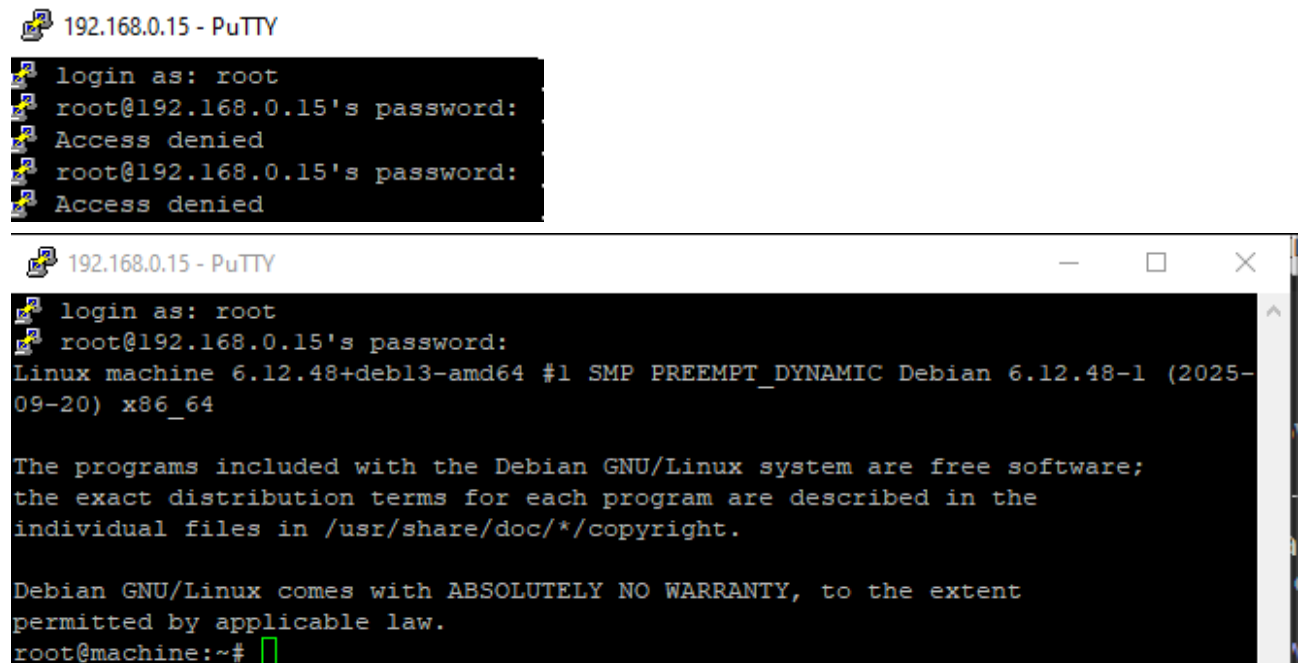
# Exercici1

Posa l'adaptador 1 de la teva MV Debian CLI en mode pont, inicia-la i instal·la `openssh-server`.

Mira la ip i intenta connectar-te, des d'un linux amb la comanda `ssh user@IP` o des del teu windows amb [putty](#). Crida al professor si no l'has fet mai.

**NOTA:** si intentem fer login com a root no funcionarà, hauríem de canviar el paràmetre `PermitRootLogin yes` del fitxer `/etc/ssh/sshd_config`

Després hem de reiniciar el servei per aplicar els canvis `systemctl restart sshd`



The image shows two screenshots of a PuTTY terminal window titled "192.168.0.15 - PuTTY". The first screenshot shows a failed login attempt where the user tries to log in as 'root' but is denied access twice. The second screenshot shows a successful login as 'root' on a Debian 6.12.48-1 system. The terminal output for the successful login includes the system version, kernel, and a warning about the GNU/Linux warranty.

```
login as: root
root@192.168.0.15's password:
Access denied
root@192.168.0.15's password:
Access denied

login as: root
root@192.168.0.15's password:
Linux machine 6.12.48+deb13-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.12.48-1 (2025-09-20) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@machine:~#
```

Un cop comprovada la connexió, ja hauries de poder copiar/enganxar les comandes mitjançant ssh.

**Abans de començar fes una instantània que es digui AbansDeApache. A la pràctica següent ens farà falta tornar a aquest punt.**

Adjunta una captura de que has fet la instantània.

Llegeix la web d'Apache de mauriciomatamala que us paso dins del [.zip](#) i realitza els exercicis adaptant els seus llocs webs a [www.cicles.ins](#) i [www.ciutat.net](#)

Fes un manual de configuració i entrega els arxius configurats si és el cas.

## Exercici2

fes que al lloc cicles demani autenticació mitjançant .htaccess per als usuaris cicles1 i cicles2, i que aquests puguin fer loguin.

Ha de funcionar amb l'àlies cicles.ins sense les www. Fes el mateix per al lloc ciutat però des de la configuració del seu HostVirtual però pels usuaris ciutat1 i ciutat2. Comprova que amb els usuaris ciutat no pots accedir a cicles i a l'inrevés.

Fes un manual de configuració i entrega els arxius configurats.

## Exercici3 webmin

Entra al teu webmin <https://localhost:10000> (user root pass root si no l'has canviat) i a dins de l'apartat server i seleccionant Apache Webserver has de tenir tot el que has configurat mitjançant els fitxers. Fes una documentació de com faries la configuració que has fet als exercicis anteriors mitjançant Webmin.

Aquí tens un manual de webmin

[https://pssp.app.br/infraestrutura/linux/servidor-web-apache/como\\_usar\\_webmin\\_para\\_configurar\\_apache2.html](https://pssp.app.br/infraestrutura/linux/servidor-web-apache/como_usar_webmin_para_configurar_apache2.html) i un pdf de com es configura apache des de webmin

(desactualitzat) <http://marigc79.files.wordpress.com/2012/12/webmin-y-apache.pdf>

i com instal·lar webmin i certificats SSL

<https://www.ssldragon.com/es/how-to/install-ssl-certificate/webmin/>

## Exercici4 SSL Opcional

Durant la pandèmia moltes organitzacions van començar a fer teletreball, i per no haver d'instal·lar cap software als ordinadors dels usuaris a casa seva, van optar per muntar un servidor amb guacamole, que el seu client és un navegador web i funciona tant per SSH com per RDP.

NOTA:Entrega la documentació de la instal·lació dels paquets necessaris, generació dels certificats i la configuració. Mostra una prova de què ha funcionat.

3.1 Que permet guacamole?

3.2 Que es SSH? i RDP?

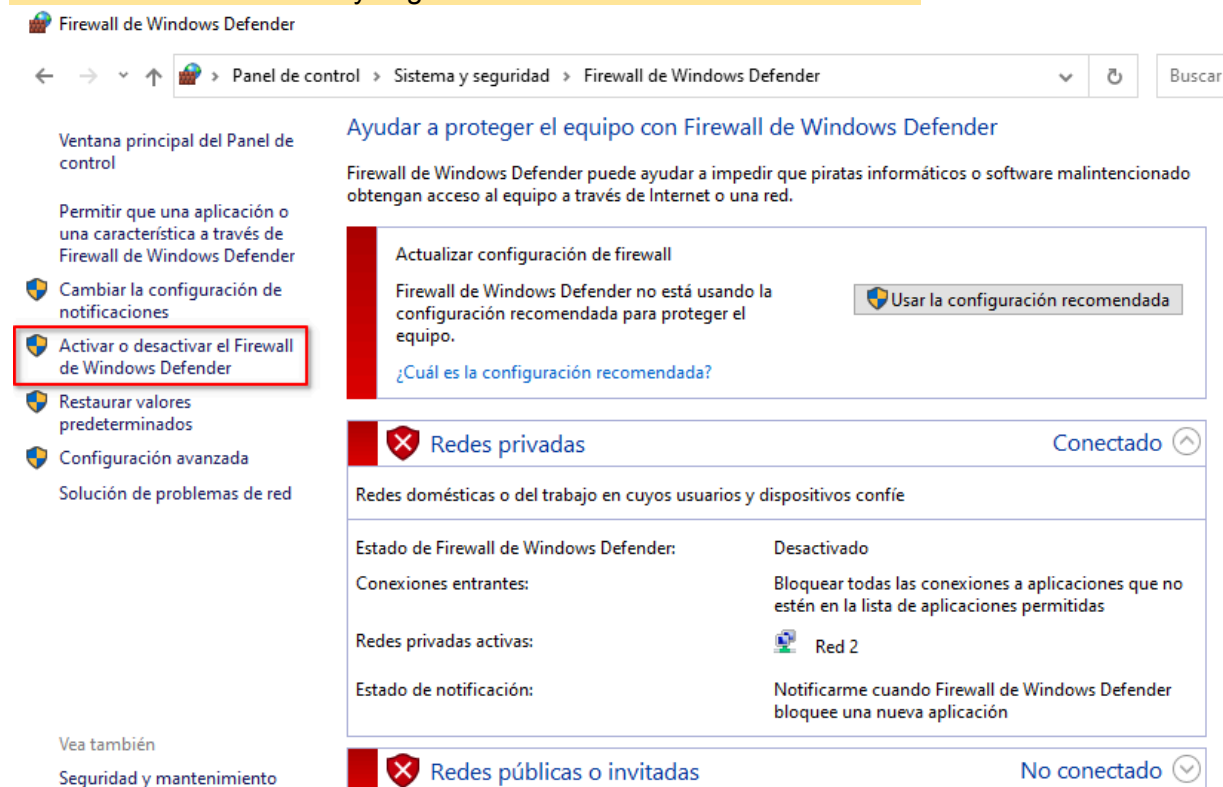
3.3 Segueix aquest manual per implementar-lo però amb l'has d'intentar adaptar per controlar la versió de debian actual i la versió de windows client si és possible.

**NOTA:** La pràctica està provada amb Virtualbox amb un UbuntuServer20.04 (que fa de servidor Guacamole), un debian 11 (provarem la connexió SSH de Guacamole) i un windows 10 (provarem la connexió RDP de Guacamole). Pots fer servir el servidor que vulguis (compte amb la versió de tomcat que instal·la)

**NOTA2:** Les màquines virtuals que faran de client no han d'estar enceses alhora si no tenim prou RAM

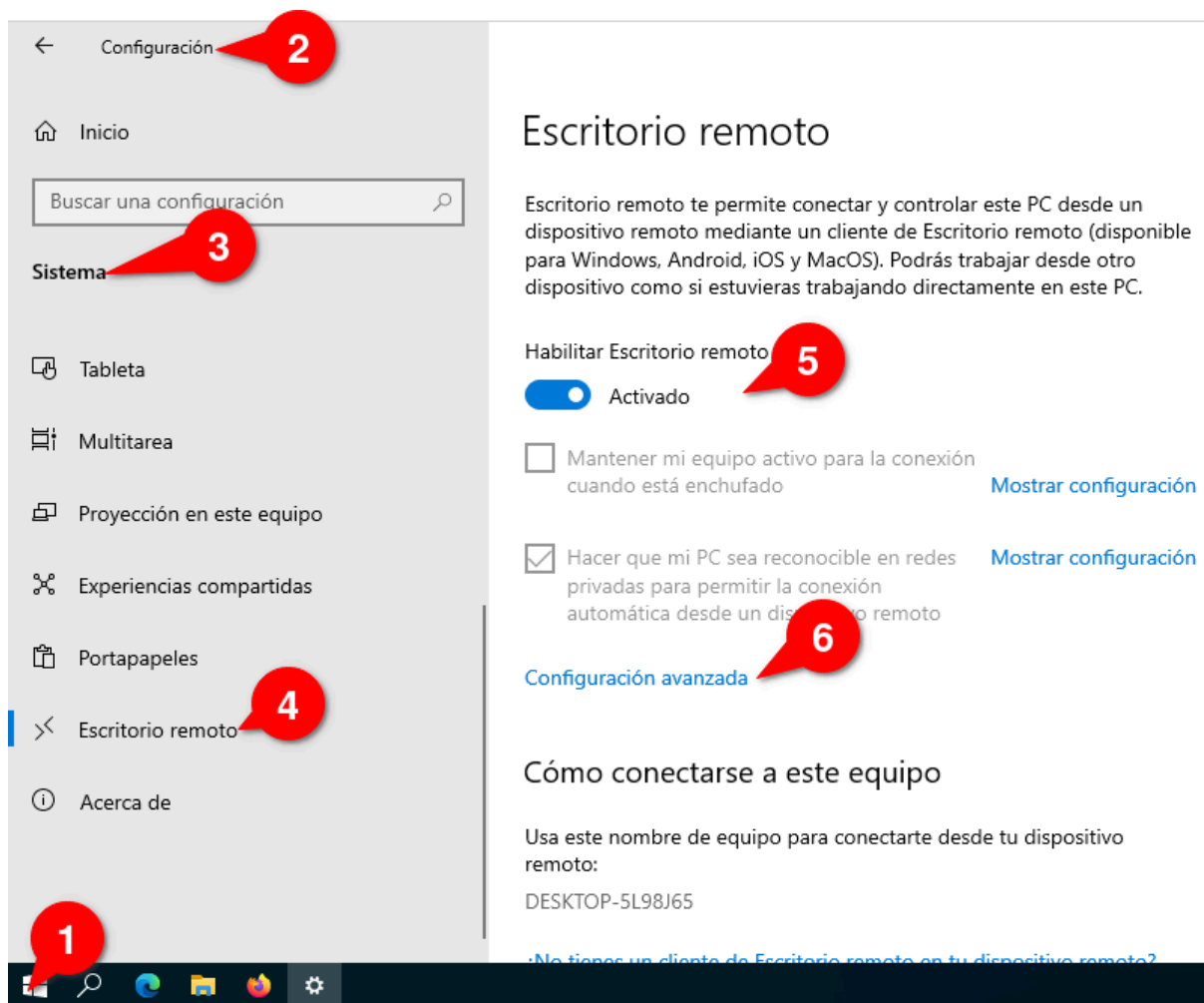
**NOTA 3:** La màquina virtual Debian client, ha de tenir instal·lat openssh-server (deixarem la configuració per defecte)

**NOTA 4:** La màquina virtual Windows com és un entorn de proves deshabilitarem el firewall "Panel de control\Sistema y seguridad\Firewall de Windows Defender"

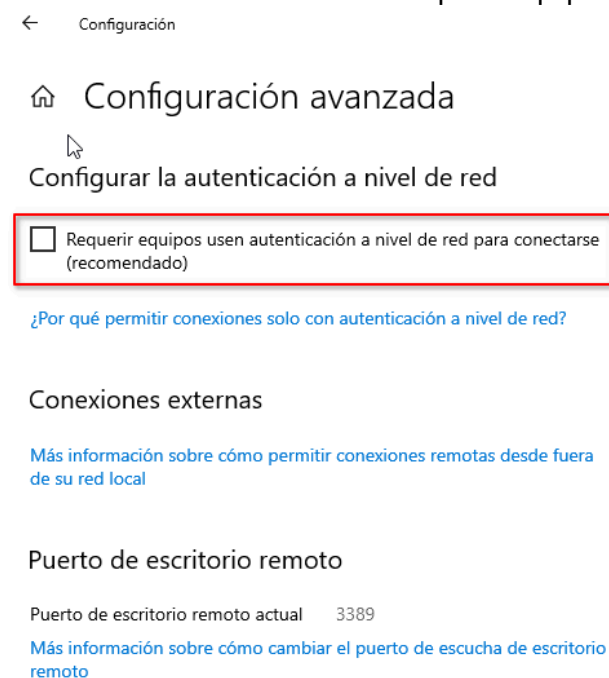


**NOTA 5:** La màquina virtual Windows hem d'activar el control remot i treure les claus segures a la part avançada

"clic Inicio/Configuración/Sistema/Scroll avall fins arribar a Escritorio Remoto/Habilitar Escritorio Remoto/ Clic Configuración Avanzada"



traiem el check de la casella “Requerir equipos usen autenticación a nivel de red para conectarse”



**L'esquema de connexió serà el següent:**

IP del HOST -> DHCP

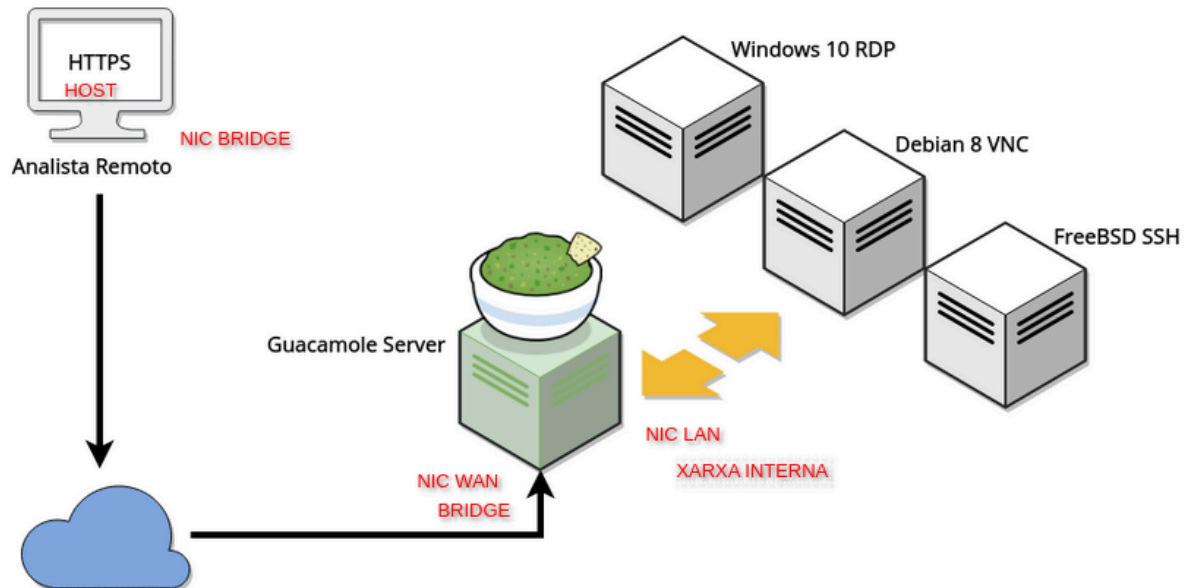
IP del servidor UBUNTU (NIC en bridge) -> Si el deixem per defecte serà DHCP, per saber la ip hem d'executar la comanda \$ip -4 a

Penseu que si fem tot per DHCP, les IPs canviaran. Les haurem de mirar cada cop que reiniciem una màquina.

IP del client debian (NIC en bridge)

IP del client windows (NIC en bridge)

Per aquesta pràctica, **els usuaris de les màquines que agafarem remotament seran usuaris locals**. No farem servir usuaris del tipus [user@gmail.com](mailto:user@gmail.com) per windows ni res per l'estil, encara que també suporta autenticació LDAP.



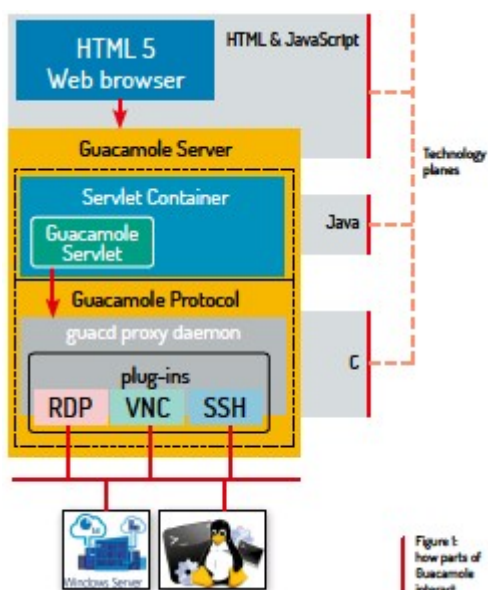
Faig servir aquests links com a guia per a realitzar la instal·lació i la configuració del servidor, però nosaltres farem servir les instruccions provades a l'exercici 1

<https://blackhold.nusepas.com/2021/03/24/instalacion-y-configuracion-de-apache-guacamole-en-debian-10/>

<https://kifarunix.com/install-guacamole-on-debian-11/>

<https://www.youtube.com/watch?v=lktlNxpOQAE> o al [drive](#)

<https://guacamole.apache.org/doc/gug/installing-guacamole.html#>



## Exercici 1. Instal·lació

Aquesta Pràctica es farà en grups de 2 alumnes.

Les IPs assignades a cada màquina la pots trobar a:

<https://docs.google.com/spreadsheets/d/1cwp8ozJurJgTTjy5uFhzRtZAHUwHS7QJiIXALQ8n3iY/edit?usp=sharing>

<https://guacamole.apache.org/releases/> aquí trobarem el manual d'instal·lació oficial i el paquet guacamole a descarregar més recent.

Aquí deixo les comandes fetes servir per funcionar Guacamole 1.4 a ubuntu [historyGuacamole1.4UbuntuServer20.04.txt](#) UN COP FINALITZADES TOTES LES COMANDES PER PROVAR FAREM

<http://server-IP:8080/guacamole> podem connectar al nostre servidor des del nostre Host fent servir el navegador i substituint server-IP per la IP del nostre servidor

**el contingut del fitxer /etc/guacamole/user-mapping.xml és: Hem d'adaptar les nostres IPs i l'usuari i el password si hem fet un de diferent (el Mill**

<user-mapping>

<!-- Per-user authentication and config information -->

<!-- A user using md5 to hash the password

guacadmin user and its md5 hashed password below is used to  
login to Guacamole Web UI-->

<authorize

username="guacadmin"

password="82af7fd93573df7b229a3867c80fd72f"

encoding="md5">

<!-- First authorized Remote connection -->

<connection name="Debian11SSH">

<protocol>ssh</protocol>

<param name="hostname">192.168.1.10</param>

<param name="port">22</param>

<!-- <param name="username">user</param> -->

</connection>

<!-- Second authorized remote connection -->

<connection name="Windows 10 RDP">

<protocol>rdp</protocol>

<param name="hostname">192.168.1.132</param>

<param name="port">3389</param>

<!-- <param name="username">user</param> -->


<param name="ignore-cert">true</param>

</connection>

</authorize>

</user-mapping>

Si volem provar qualsevol versió de guacamole i compilar-lo. El podem fer amb git clone, però com no hem fet git farem servir wget.

[guacamole-client-1.4.0.tar.gz](#) [ [PGP](#) ] [ [SHA-256](#) ]  
[guacamole-server-1.4.0.tar.gz](#) [ [PGP](#) ] [ [SHA-256](#) ]  [Clic dret, copia enllaç](#)

#wget “enganxa l’enllaç copiat”

## Entrega:

Hem d’entregar el link del vídeo que farem amb OBS, de com fem clic al nostre guacamole i ens connectem remotament al Nostre debian i al nostre windows. Si és possible crida al professor i li ensenyes a classe.

Fes una llista amb les comandes que no entenguis del fitxer que us he passat, i explica que fan. Com a mínim has d’explicar 8 comandes. Pots consultar aquesta web que explica moltes <https://www.ochobitshacenunbyte.com/2020/10/21/administracion-remota-con-apache-guacamole/>

## Exercici 2: HTTPS

### 0. Crea SnapShot

Crea un Snapshot que es digui AbansDeHttps

### 1.- Generarem un certificat autosignat per a SSL

<https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-18-04-es>

**Nota:** Un certificat autosignat xifrarà la comunicació entre el vostre servidor i qualsevol client. No obstant això, com que no està signat per cap de les autoritats certificadores de confiança incloses amb els navegadors web, els usuaris no poden utilitzar el certificat per validar la identitat del vostre servidor de forma automàtica.

És possible que un certificat autosignat sigui apropiat si no disposeu d’un nom de domini associat amb el vostre servidor i per a casos en què una interfície web xifrada no estigui dirigida a l’usuari. Si disposeu d’un nom de domini, en molts casos és millor fer servir un certificat signat per una autoritat certificadora (CA). Podeu esbrinar la manera de configurar un certificat de confiança gratuït a través del projecte [Let’s Encrypt aquí](#).



A més farà de [proxy invers](#), ens protegirà de les connexions que arribin des d'Internet.

<https://www.bujarra.com/instalando-apache-guacamole/> (ssl amb apache)

### 1.1.- Comprovem que tenim instal·lat openssl i quina versió

```
# openssl version -a
```

podem fer una ullada a totes les comandes openssl amb

```
#openssl help
```

### 1.2.- consulta la ip del teu servidor, que farà falta després

```
#ip -4 a
```

### 1.3.- Crearem la key i el certificat SSL necessaris

```
# openssl req -x509 -nodes -days 1000 -newkey rsa:4096 -keyout
```

```
/etc/ssl/private/aalcala-selfsigned.key -out /etc/ssl/certs/aalcala-selfsigned.crt
```

### posa la ip anterior com a resposta al FQDN (quadrat vermell de la captura)

```
user@server:~$ sudo openssl req -x509 -nodes -days 1000 -newkey rsa:4096 -keyout /etc/ssl/private/aalcala-selfsigned.key -out /etc/ssl/certs/aalcala-selfsigned.crt
Generating a RSA private key
.....+++++
writing new private key to '/etc/ssl/private/aalcala-selfsigned.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Catalunya
Locality Name (eg, city) []:Palamos
Organization Name (eg, company) [Internet Widgits Pty Ltd]:InsPalamos
Organizational Unit Name (eg, section) []:Cicles
Common Name (e.g. server FQDN or YOUR name) []:192.168.1.140
Email Address []:aalcala@inspalamos.cat
user@server:~$
```

analitzem la comanda:

**openssl:** és l'eina de línia d'ordres bàsica per crear i administrar certificats, claus i altres fitxers d'OpenSSL.

**req:** aquesta subcomanda especifica que volem utilitzar l'administració de la sol·licitud de signatura de certificats (CSR) X.509. El "X.509" és un estàndard d'infraestructura de claus públiques al qual s'adequa SSL i TLS per a l'administració de claus i certificats a través d'aquest. Volem crear un nou certificat X.509, per la qual cosa farem servir aquesta subcomanda.

**-x509:** modifica encara més el subordre anterior en indicar a la utilitat que desitgem crear un certificat autosignat en lloc de generar una sol·licitud de signatura de certificats, com normalment succeeix.

**-nodes:** indica a OpenSSL que ometi l'opció per protegir el nostre certificat amb una frase de contrasenya. Necessitem que Apache pugui llegir el fitxer, sense intervenció de l'usuari, quan s'iniciï el servidor. Una frase de contrasenya evitaria que això passi perquè l'hauríem d'ingressar després de cada reinici.

**-days 1000:** aquesta opció estableix el temps durant el qual el certificat es considerarà vàlid. En aquest cas, ho configurem per un any.

**-newkey rsa:4096:** especifica que volem generar un nou certificat i una nova clau alhora. No creem la clau que es requereix per signar el certificat en un pas anterior, per la qual cosa l'hem de crear juntament amb el certificat. La part rsa:2048 us indica que creeu una clau RSA de 2048 bits d'extensió.

**-keyout:** aquesta línia indica a OpenSSL on col·locar el fitxer de clau privada generat que estem creant.



**-out:** indica a OpenSSL on col·locar el certificat que creem.

## 2. Configurem apache per a SSL

### 2.1. Instal·lem apache

```
#apt install apache2
```

### 2.2 creem un fitxer amb un conjunt de paràmetres per la configuració segura de SSL

```
# nano /etc/apache2/conf-available/ssl-params.conf
```

```
SSLCipherSuite ECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
SSLProtocol All -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
SSLHonorCipherOrder On
# Disable preloading HSTS for now. You can use the commented out header line that includes
# the "preload" directive if you understand the implications.
# Header always set Strict-Transport-Security "max-age=63072000; includeSubDomains; preload"
Header always set X-Frame-Options DENY
Header always set X-Content-Type-Options nosniff
# Requires Apache >= 2.4
SSLCompression off
SSLUseStapling on
SSLStaplingCache "shmcb:logs/stapling-cache(150000)"
# Requires Apache >= 2.4.11
SSLSessionTickets Off
```

### 2.3 realitzarem una còpia de seguretat del fitxer original de host virtual de SSL i després el modificarem

```
# cp /etc/apache2/sites-available/default-ssl.conf /etc/apache2/sites-available/default-ssl.conf.bak
```

```
# nano /etc/apache2/sites-available/default-ssl.conf
```

```
<IfModule mod_ssl.c>
    <VirtualHost _default_:443>
        ServerAdmin your\_email@example.com
        ServerName server\_domain\_or\_IP

        DocumentRoot /var/www/html

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        SSLEngine on

        #SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
        #SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

        SSLCertificateFile /etc/ssl/certs/aalcala-selfsigned.crt
        SSLCertificateKeyFile /etc/ssl/private/aalcala-selfsigned.key

        <FilesMatch "\.(cgi|shtml|phtml|php)$">
```

```

                SSLOptions +StdEnvVars
</FilesMatch>
<Directory /usr/lib/cgi-bin>
                SSLOptions +StdEnvVars
</Directory>

</VirtualHost>
</IfModule>

```

## 2.4 redireccionem el tràfic http a https

# nano /etc/apache2/sites-available/000-default.conf

```

<VirtualHost *:80>
    ...

    Redirect "/" "https://192.168.1.140/"

    ...
</VirtualHost>

```

## 3.- El punt 3 (no cal per la pràctica, per producció sí)

### Habilitem Firewall

# sudo ufw enable

### polítiques per defecte

#sudo ufw default deny incoming

#sudo ufw default allow outgoing

### permetre protocols coneguts

#sudo ufw allow ssh

#sudo ufw allow http

#sudo ufw allow https

### altres comandes útils ->

permetre X11 (si no especifiquem protocol al interval de ports aplica tant a udp com a tcp)

#sudo ufw allow 6000:6007/tcp

#sudo ufw allow 6000:6007/udp

### permetre un equip determinat ->

#sudo ufw allow from IP-Equip/CIDR

també admet només ip a un port o subxarxa a un port

#sudo ufw allow from 203.0.113.4 to any port 22

### més a ->

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-20-04-es>

## 3.1 Mirem aplicacions disponibles del firewall

```
#sudo ufw app list
```

a mi em surten aquestes

Available applications:

Apache

Apache Full

Apache Secure

OpenSSH

### 3.2 Consultem les que estan aplicades

```
#sudo ufw status numbered
```

### 3.3 deshabilitem Apache i habilitem Apache Full

```
#sudo ufw allow 'Apache Full'
```

```
#sudo ufw delete allow 'Apache'
```

## 4.- Habilitem canvis Apache

### 4.1 habilitem els mòduls i encapçalats SSL d'Apache

```
#sudo a2enmod ssl
```

```
#sudo a2enmod headers
```

### 4.2 Habilitem el nostre host virtual llest per a SSL, amb els paràmetres configurats al punt 2.3

```
#sudo a2ensite default-ssl
```

```
#sudo a2enconf ssl-params
```

**comprovem que no hagi errors de sintaxi abans de reiniciar amb**

```
#sudo apache2ctl configtest
```

```
user@server:~$ sudo ufw status
[sudo] password for user:
Status: inactive
user@server:~$ sudo a2enmod ssl
[sudo] password for user:
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
user@server:~$ sudo a2enmod headers
Enabling module headers.
To activate the new configuration, you need to run:
  systemctl restart apache2
user@server:~$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
user@server:~$ sudo a2enconf ssl-params
Enabling conf ssl-params.
To activate the new configuration, you need to run:
  systemctl reload apache2
user@server:~$ sudo apache2ctl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
Syntax OK
```

### 4.3 Reiniciem Apache

```
#sudo systemctl restart apache2
```

## 5.- Provem certificat

podem escriure [http://IP\\_Server](http://IP_Server) al navegador i provem la redirecció a https i els certificats alhora.

Haurem d'acceptar els certificats, ja que els hem autosignat i no els fem servir des d'una [CA](#) pública

https://192.168.1.140

Advertencia: riesgo potencial de seguridad a continuación

Firefox ha detectado una posible amenaza de seguridad y no ha cargado 192.168.1.140. Si visita este sitio, los atacantes podrían intentar robar información como sus contraseñas, correos electrónicos o detalles de su tarjeta de crédito.

[Más información...](#)

[Retroceder \(recomendado\)](#) [Avanzado...](#)

192.168.1.140 usa un certificado de seguridad no válido.

No se confía en el certificado porque está autofirmado.

Código de error: [MOZILLA\\_PKIX\\_ERROR\\_SELF\\_SIGNED\\_CERT](#)

[Ver certificado](#)

[Retroceder \(recomendado\)](#) [Aceptar el riesgo y continuar](#)

2

https://192.168.1.140

**Apache2 Ubuntu Default Page**

**ubuntu**

**It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

6.- redirecció http cap a https de manera permanent, per no fer servir el protocol http que és insegur

tornem a editar el fitxer

```
#sudo nano /etc/apache2/sites-available/000-default.conf
```

modificant la línia d'abans. Ha de quedar

Redirect **permanent** "/" "https://192.168.1.140/"

```

<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    Redirect permanent "/" "https://192.168.1.140/"
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

```

comprovem la sintaxi i si tot és correcte reiniciem apache i tornem a provar des del navegador

```

#sudo apache2ctl configtest
#sudo systemctl restart apache2

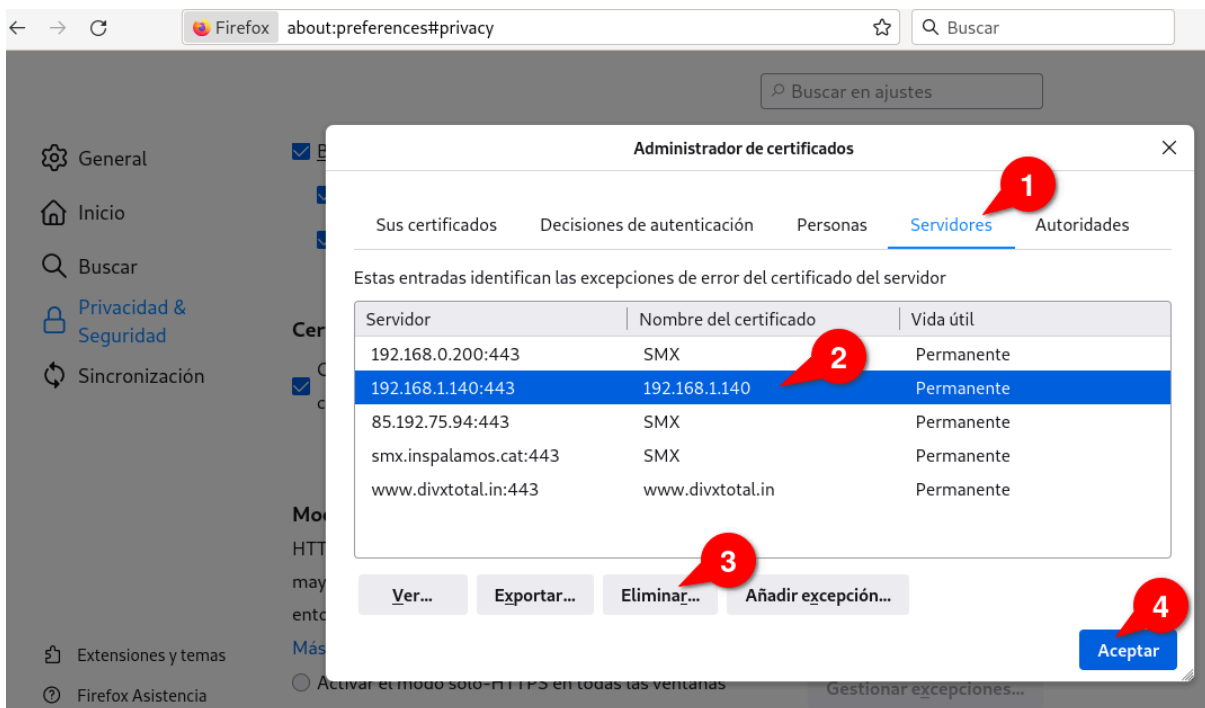
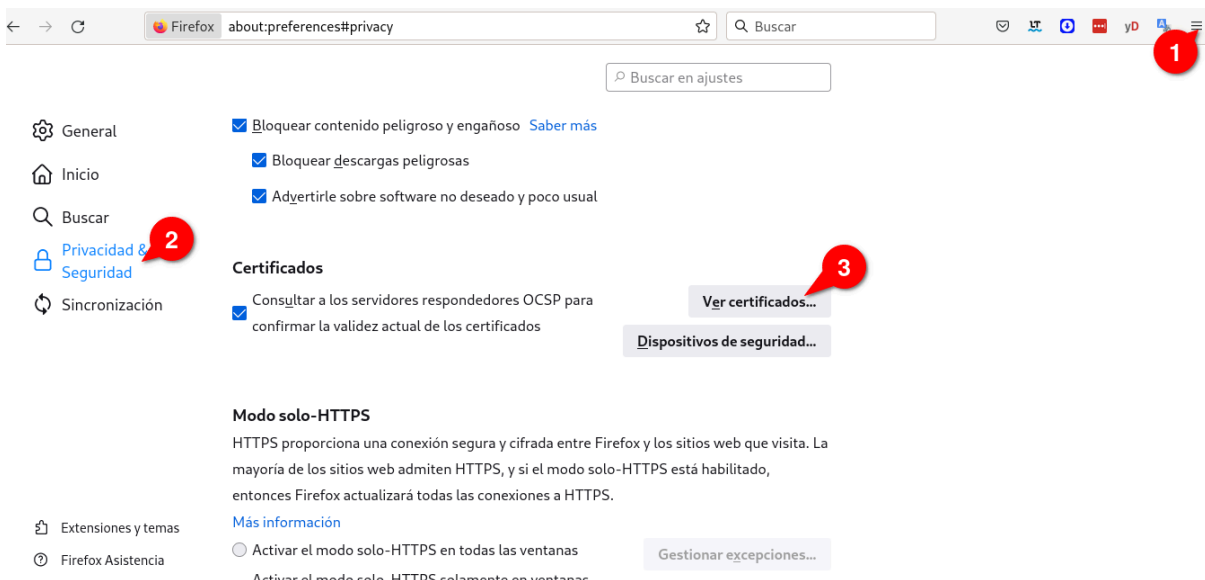
```

## NOTA:

Si en algun moment hem d'eliminar el nostre certificat de Firefox anirem a Ajustes-> Privacidad & Seguridad -> Certificados -> Ver Certificados

a la finestra que ens mostra anem a la pestanya servidores -> seleccionem el certificat a eliminar -> eliminar -> aceptar

Ara si tornem a ficar la nostra IP al navegador ens tornarà a demanar que acceptem el certificat, com avans, ja que l'estem certificant nosaltres mateixos.



## 7.- Habilitem guacamole per SSL

Ara que hem provat que els certificats funcionem anem a dir-li a guacamole que els faci servir

### 7.1 habilitem els mòduls de proxy d'apache

```
# sudo a2enmod proxy proxy_http proxy_balancer
```

```
user@server:~$ sudo a2enmod proxy proxy_http proxy_balancer
[sudo] password for user:
Enabling module proxy.
Considering dependency proxy for proxy_http:
Module proxy already enabled
Enabling module proxy_http.
Considering dependency proxy for proxy_balancer:
Module proxy already enabled
Considering dependency alias for proxy_balancer:
Module alias already enabled
Considering dependency slotmem_shm for proxy_balancer:
Enabling module slotmem_shm.
Enabling module proxy_balancer.
To activate the new configuration, you need to run:
    systemctl restart apache2
user@server:~$ █
```

## 7.2 editem el fitxer

```
#sudo nano /etc/apache2/sites-available/default-ssl.conf
```

afegim aquest contingut (mira la captura)

```
SSLProxyEngine on
SSLProxyVerify none
SSLProxyCheckPeerCN off
SSLProxyCheckPeerName off
SSLProxyCheckPeerExpire off
<Location /guacamole/>
Order allow,deny
Allow from all
ProxyPass http://DIRECCION_IP_GUACAMOLE:8080/guacamole/ flushpackets=on
ProxyPassReverse http://DIRECCION_IP_GUACAMOLE:8080/guacamole/
</Location>
```



```

GNU nano 4.8 /etc/apache2/sites-available/default-ssl.conf
SSLEngine on

# A self-signed (snakeoil) certificate can be created by installing
# the ssl-cert package. See
# /usr/share/doc/apache2/README.Debian.gz for more info.
# If both key and certificate are stored in the same file, only the
# SSLCertificateFile directive is needed.
#SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
#SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

SSLCertificateFile /etc/ssl/certs/aalcala-selfsigned.crt
SSLCertificateKeyFile /etc/ssl/private/aalcala-selfsigned.key

#####SSL GUACAMOLE https://www.bujarra.com/instalando-apache-guacamole/
SSLProxyEngine on
SSLProxyVerify none
SSLProxyCheckPeerCN off
SSLProxyCheckPeerName off
SSLProxyCheckPeerExpire off
<Location /guacamole/>
    Order allow,deny
    Allow from all
    ProxyPass http://192.168.1.140:8080/guacamole/ flushpackets=on
    ProxyPassReverse http://192.168.1.140:8080/guacamole/
</Location>

# Server Certificate Chain:
# Point SSLCertificateChainFile at a file containing the
# concatenation of PEM encoded CA certificates which form the

```

### 7.3 Comprovem sintaxi, reiniciem i provem

#sudo apache2ctl configtest

#sudo systemctl restart apache2

Escrivim al navegador [https://IP\\_Servidor/guacamole/](https://IP_Servidor/guacamole/)

← → ↻

Q https://192.168.1.140/guacamole/

C



**APACHE GUACAMOLE**

...

...

Iniciar Sesión

## Entrega:

Hem d'entregar el link del vídeo que farem amb OBS, de com fem clic al nostre guacamole per https i ens connectem remotament al Nostre debian i al nostre windows. Si és possible crida al professor i li ensenyes a classe.

Si vols personalitzar la pàgina de login de guacamole pots seguir el manual

<https://www.bujarra.com/tema-corporativo-en-apache-guacamole/>

## Pràctica Opcional 1 Guacamole

Torna al Snapshot que has fet abans de Htpps i crea un altre de l'estat actual que es digui Guacamole\_HTTPS\_ApacheOK

Fes el mateix que hem fet amb apache però amb nginx ->

<https://www.youtube.com/watch?v=Yz5bZl4xqgg>

<https://www.ochobitshacenunbyte.com/2019/08/21/instalar-y-configurar-un-proxy-inverso-con-nginx-en-ubuntu-18-04/>

<https://www.ochobitshacenunbyte.com/2020/10/21/administracion-remota-con-apache-guacamole/>  
(ssl amb nginx, certbot i lets Encrypt)

Si funciona crea un SnapShot que es digui Guacamole\_HTTPS\_NginxOK

## Pràctica Opcional 2 Guacamole Integració amb Active Directory -> NO S'HA DE FER

Si no has fet la pràctica Opcional de Nginx, crea un SnapShot de l'estat actual de la MV que es digui Guacamole\_HTTPS\_ApacheOK. Si l'has fet restaura l'SnapShot amb el nom Guacamole\_HTTPS\_ApacheOK

Investiga com fer que usuaris d'Active Directory puguin fer servir Guacamole. Documenta tot el procés. No cal la part d'instal·lació dels Sistemes Operatius ni de posar en marxa AD. De la part de windows server només fa falta una captura d'un parell d'usuaris creats. L'important és la configuració de Guacamole perquè assigni a cada usuari AD al client windows assignat.

Per fer servir usuaris d'active Directory podem provar aquest manual

<https://www.bujarra.com/autenticacion-de-apache-guacamole-contra-directorio-activo/>

i veure aquest vídeo

<https://www.youtube.com/watch?v=AKhgSLjQk38>

Pots demanar un Windows Server al professor (només hauràs d'instal·lar la característica AD, promocionar-lo a domini, crear 2 usuaris i afegir el client windows a domini. Un cop vegis que un usuari pot accedir i l'altre no, treu el client del domini i aprofita per canviar-li el nom. El trones a unir a domini i aquí és on ha de poder connectar-se l'usuari que abans no podia i el que podia no s'ha de poder connectar a aquesta màquina)

NOTA: per fer login a windows Server → usuari Administrador i Contrasenya: P@ssw0rd

