Administració de servidors

Remo Suppi Boldrito

PID_00212465

Universitat Oberta de Catalunya

Índex

Introducció

La interconnexió entre màquines i les comunicacions d'alta velocitat han permès que els recursos que s'utilitzin no estiguin al mateix lloc geogràfic de l'usuari. UNIX (i per descomptat, GNU/Linux) és probablement el màxim exponent d'aquesta filosofia, ja que des del seu inici ha fomentat l'intercanvi de recursos i la independència de dispositius. Aquesta filosofia s'ha plasmat en una cosa comuna avui dia com són els serveis. Un servei és un recurs (que pot ser universal o no) que permet, sota certes condicions, obtenir informació, compartir dades o simplement processar la informació a distància. El nostre objectiu és analitzar els serveis que permeten el funcionament d'una xarxa. Generalment, dins d'aquesta xarxa hi haurà una màquina (o més, segons les configuracions) que farà possible l'intercanvi d'informació entre les altres. Aquestes màquines es denominen *servidors* i contenen un conjunt de programes que permeten que la informació estigui centralitzada i sigui fàcilment accessible. Aquests serveis permeten la reducció de costos i amplien la disponibilitat de la informació, però s'ha de tenir en compte que un servei centralitzat presenta inconvenients, ja que pot quedar fora de servei i deixar sense atenció tots els usuaris. En aquest mòdul, es veuran els principals serveis que permeten que una màquina GNU/Linux jugui un paper molt important en una infraestructura tecnològica, tant a centralitzar i distribuir dades com a ser punt d'informació, accés o comunicació. D'altra banda, i amb l'avenç de les arquitectures (programari) orientades a serveis (SOA - *Service Oriented Architecture*), i les tecnologies de desenvolupament d'aplicacions que s'han estandarditzat en aquest paradigma de disseny de sistemes distribuïts, GNU/Linux s'ha transformat en la infraestructura per excel·lència que dóna suport a la creació de sistemes d'informació altament escalables. Aquest tipus d'arquitectura (SOA) s'ha transformat en una part essencial del desenvolupament de programari distribuït, ja que permet la creació de sistemes distribuïts eficients, que aprofiten tota la infraestructura subjacent, i estableix una interfície ben definida a l'exposició i crida de serveis web (de manera comuna però no exclusivament), cosa que facilita la interacció entre els sistemes propis i externs.

Serveis replicats

Una arquitectura de servidors ha de tenir els serveis replicats *(mirrors)* per a solucionar els inconvenients que comporta.

Objectius

En els materials didàctics d'aquest mòdul, trobareu els continguts i les eines procedimentals per a aconseguir els objectius següents:

1. Presentar els aspectes més rellevants dels conceptes involucrats, tant en un nivell teòric com pràctic, en l'estructura de servidors/serveis en un sistema GNU/Linux.
2. Analitzar els conceptes relatius a serveis i servidors específics d'un sistema GNU/Linux.
3. Experimentar amb la configuració i adaptar la instal·lació de serveis a un entorn determinat.
4. Analitzar i participar en discussions sobre les possibilitats actuals i futures de nous serveis i els obstacles que hi ha bàsicament en aspectes de seguretat en els diferents entorns de treball GNU/Linux (servidor, escriptori multimèdia, escriptori ofimàtica, encaminador o *router*, etc.).

1. Administració de servidors

Els serveis es poden classificar en dos tipus: de vinculació ordinador-ordinador o de relació home-ordinador. En el primer cas, es tracta de serveis requerits per altres ordinadors, mentre que, en el segon, són serveis requerits pels usuaris (encara que hi ha serveis que poden actuar en ambdues categories). Dins del primer tipus es troben serveis de noms, com el *Domain Name System* (DNS), el servei d'informació d'usuaris (NIS-YP), el directori d'informació LDAP o els serveis d'emmagatzematge intermedi (*proxies*). Dins de la segona categoria es preveuen serveis de connexió interactiva i execució remota (ssh, Telnet), transferència de fitxers (ftp), intercanvi d'informació en nivell d'usuari, com el correu electrònic (MTA, IMAP, POP), *news*, *World Wide Web*, *wiki* i arxius (NFS). Per a mostrar les possibilitats de GNU/Linux Debian-FC, es descriurà cadascun d'aquests serveis amb una configuració mínima i operativa, però sense descuidar aspectes de seguretat i estabilitat.

1.1. Sistema de noms de domini (*Domain Name System, DNS*)

La funcionalitat del servei de DNS és convertir noms de màquines (llegibles i fàcils de recordar pels usuaris) en adreces IP o viceversa.

A la consulta de quina és la IP de *nteum.remix.cat*, el servidor respondrà 192.168.0.1 (aquesta acció és coneguda com a *mapping*); de la mateixa manera, quan se li proporcioni l'adreça IP, respondrà amb el nom de la màquina (conegut com *reverse mapping*).

El *Domain Name System* (DNS) és una arquitectura arborescent que evita la duplicació de la informació i facilita la cerca. Per això, un únic DNS només té sentit com a part de l'arbre. Una de les aplicacions més utilitzades que presta aquest servei es diu *named*, s'inclou en la majoria de distribucions de GNU/Linux (`/usr/sbin/named`) i forma part d'un paquet anomenat `bind` (actualment versió 9.x), coordinat per l'ISC (Internet Software Consortium). El DNS és simplement una base de dades, per la qual cosa, és necessari que les persones que la modifiquin coneguin la seva estructura, ja que, en cas contrari, el servei quedarà afectat. Com a precaució, ha de tenir-se especial cura a guardar les còpies dels arxius per a evitar qualsevol interrupció en el servei. Els servidors DNS que poden convertir la majoria de nodes de DNS en la seva IP corresponent i es denominen *servidors DNS recursius*. Aquest tipus de

servidor no pot canviar els noms dels nodes de DNS que hi ha, simplement es pregunta a altres servidors DNS la IP d'un node DNS donat. Els servidors DNS autoritzats poden gestionar/canviar les adreces IP dels nodes DNS que gestionen i normalment són amb els quals contactarà un servidor DNS recursiu amb la finalitat de conèixer la IP d'un node DNS determinat. Una tercera variant dels servidors DNS són els DNS cau, que simplement emmagatzemen la informació obtinguda d'altres servidors DNS recursius.

1.1.1. Servidor de noms cau

En primer lloc, es configurarà un servidor de DNS per a resoldre consultes, que actuï com a cau per a les consultes de noms *resolver, caching only server*). És a dir, la primera vegada consultarà el servidor adequat perquè es parteix d'una base de dades sense informació, però les vegades següents respondrà el servidor de noms cau, amb la corresponent disminució del temps de resposta. Per a instal·lar un servidor d'aquestes característiques, instal·lem els paquets `bind9 i dnsutils` (p. ex., en Debian `apt-get bind9 dnsutils`).

Com es pot observar en el directori */etc/bind*, trobarem una sèrie d'arxius de configuració, i entre aquests el principal és *named.conf*, que inclou altres *named.conf.options, named.conf.local, named.conf.default-zones* (podem trobar que aquesta configuració pot variar entre distribucions, però és similar). L'arxiu */etc/bind/named.conf.options* conté les opcions generals per al servidor i els altres dos arxius ens serviran per a configurar les nostres zones en el servidor de noms que veurem en l'apartat següent. Per a configurar el nostre *caching only server*, hem de considerar que les preguntes en primer lloc les resoldrem nosaltres, però com que és evident que no tenim la resposta, s'haurà de redirigir la pregunta a servidors de DNS en la Xarxa. Per a això, és interessant considerar els de serveis com OpenDNS* que són les IP: 208.67.222.222 i 208.67.220.220 o bé serveis com els de Google** que responen a les IP: 8.8.8.8 i 8.8.4.4. A continuació, modifiquem l'arxiu */etc/bind/named.conf.options* per a treure els comentaris de la secció *forwarders* posant el següent:

***http://www.opendns.com /opendns-ip-addresses/**
****https://developers.google.com /speed/public-dns/?csw=1**

```
forwarders {
   // OpenDNS servers
    208.67.222.222;
    208.67.220.220;
   // Podríem incloure el nostre ISP/router -verificar la IP-
    192.168.1.1;
   };
```

En el mateix arxiu al final i reemplaçant les que hi ha (es pot deixar aquesta configuració per a un segon pas), es podrien agregar opcions de seguretat perquè només resolgui les peticions de la nostra xarxa:

```
// Security options
 listen-on port 53 { 127.0.0.1; 192.168.1.100; };
 allow-query { 127.0.0.1; 192.168.1.0/24; };
```

```
allow-recursion { 127.0.0.1; 192.168.1.0/24; };
allow-transfer { none; };
// Les següents dues opcions ja es troben per defecte en
// l'arxiu, però si no tenim IPv6 podem comentar l'última.
auth-nxdomain no; # conform to RFC1035
// listen-on-v6 { any; };
```

Es pot verificar la configuració amb `named-checkconf`. A continuació, modifiquem l'arxiu */etc/resolv.conf* per a afegir `nameserver 127.0.0.1` perquè les preguntes a DNS la biblioteca *gethostbyname(3)* les faci al propi *host* i l'arxiu */etc/nsswitch.conf* verificant que la línia *hosts* quedi com: `hosts: files dns` i indicarà l'ordre en què es resoldran les peticions (primer localment a */etc/hosts* i després al servidor DNS). Finalment, només resta reiniciar el servidor amb `service bind9 restart` i fer les proves de funcionament. En el nostre cas, hem executat `dig www.debian.org` (s'han omès línies per a resumir la informació):

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» www.debian.org
...
;; QUESTION SECTION:
;www.debian.org. IN A

;; ANSWER SECTION:
www.debian.org. 300 IN A 130.89.148.14
www.debian.org. 300 IN A 5.153.231.4
...
;; Query time: 213 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
...
```

On podem observar que la *query* ha trigat 213 mil·lisegons. Si la fem per segona vegada (ja ha quedat la consulta emmagatzemada en el nostre servidor cau):

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» www.debian.org
...
;; QUESTION SECTION:
;www.debian.org. IN A

;; ANSWER SECTION:
www.debian.org. 293 IN A 5.153.231.4
www.debian.org. 293 IN A 130.89.148.14
...
;; Query time: 1 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
...
```

En aquest cas, la consulta ha trigat 1 mil·lisegon a verificar que el servidor cau ha funcionat. També podem fer la consulta inversa amb la instrucció `nslookup 130.89.148.14`, la qual cosa ens donarà el nom de la màquina que té assignada aquesta IP (en aquest cas, és el servidor al qual està assignat el domini www.debian.org):

```
Server: 127.0.0.1
Address: 127.0.0.1#53
Non-authoritative answer:
```

```
14.148.89.130.in-addr.arpa name = klecker4.snt.utwente.nl.
Authoritative answers can be found from:
. nameserver = l.root-servers.net.
. nameserver = b.root-servers.net.
. nameserver = j.root-servers.net.
...
```

És important considerar si es desitja instal·lar el servidor de DNS a una màquina virtual, i tenint en compte que és un servei al qual s'ha d'accedir des de fora, el servidor DNS (i la majoria de servidors) no pot estar en NAT sobre un *host* sinó que ha de tenir IP pròpia del segment de xarxa en la qual presta el servei i, per tant, l'adaptador de xarxa de la màquina virtual ha d'estar en configuració pont (*bridged*). Per a configurar els clients dins de la nostra xarxa, n'hi haurà prou de modificar l'arxiu */etc/resolv.conf* si són GNU/Linux (o a modificar */etc/network/interfaces* per a agregar-los com `dns-nameservers IP` si es té instal·lat el paquet `resolvconf`), i en Windows a modificar les propietats IPv4 de l'adaptador per a introduir l'IP del nostre servidor DNS cau.

1.1.2. Configuració d'un domini propi

El DNS posseeix una estructura en arbre i l'origen és conegut com a "." (vegeu `/etc/bind/db.root`). Sota el "." hi ha els TLD (*Top Level Domains*) com **org**, **com**, **edu**, **net**, etc. Quan es busca en un servidor, si aquest no coneix la resposta, es buscarà de manera recursiva en l'arbre fins a trobar-la. Cada "." en una adreça (per exemple, nteum.remix.cat) indica una branca de l'arbre de DNS diferent i un àmbit de consulta (o de responsabilitat) diferent que s'anirà recorrent de manera recursiva d'esquerra a dreta.

Un altre aspecte important, a més del domini, és el `in-addr.arpa` (*inverse mapping*), el qual també està imbricat com els dominis i serveix per a obtenir noms quan es consulta per l'adreça IP. En aquest cas, les adreces s'escriuen a l'inrevés, en concordança amb el domini. Si nteum.remix.world és la 192.168.0.30, llavors s'escriurà com 30.0.168.192, en concordança amb nteum.remix.world per a la resolució inversa d'IP -> nom. En aquest exemple, utilitzarem un servidor de DNS que atén una xarxa interna (192.168.0.0/24) i després té una IP pública, però per a la finalitat de l'exemple i per a poder fer les proves internes la configurarem en una IP privada d'un altre segment (172.16.0.80) i un domini fictici (remix.world). En el cas de portar aquest servidor a producció, s'hauria de canviar per la IP externa que tingui i el domini corresponent que s'hagi obtingut i tinguem a la nostra disposició.

Per a configurar un servidor de noms propi, en primer lloc canviarem l'arxiu */etc/bind/named.conf* per a definir dues zones (una d'interna i una altra d'externa). L'arxiu quedarà com (observeu que hem comentat la línia que inclou *named.conf.default-zones*, ja que la inclourem després):

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
```

```
    //include "/etc/bind/named.conf.default-zones";
    include "/etc/bind/named.conf.internal-zones";
    include "/etc/bind/named.conf.external-zones";
```

Modifiquem l'arxiu *named.conf.internal-zones* per a indicar-li els arxius on es trobaran realment els noms de les màquines, que en aquest cas seran dos: un per a resolució nom -> IP anomenat *remix.world.lan* i un altre per a la resolució inversa IP -> nom anomenat 0.168.192.db.

```
view "internal" {
    match-clients {
        localhost;
        192.168.0.0/24;
    };
// set zone for internal
    zone "remix.world" {
        type master;
        file "/etc/bind/remix.world.lan";
        allow-update { none; };
    };
// set zone for internal reverse
    zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/0.168.192.db";
        allow-update { none; };
    };
include "/etc/bind/named.conf.default-zones";
};
```

De la mateixa manera, modifiquem el fitxer *named.conf.external-zones* per a indicar-li els arxius que resoldran aquesta zona i que també seran dos: remix.world.wan i 80.0.16.172.db per a la resolució directa i inversa, respectivament.

```
view "external" {
// define for external section
    match-clients { any; };
// allow any query
    allow-query { any; };
// prohibit recursion
    recursion no;
// set zone for external
    zone "remix.world" {
        type master;
        file "/etc/bind/remix.world.wan";
        allow-update { none; };
    };
// set zone for external reverse
    zone "80.0.16.172.in-addr.arpa" {
        type master;
        file "/etc/bind/80.0.16.172.db";
        allow-update { none; };
    };
};
```

També s'ha de modificar el *named.conf.options*:

```
options {
    directory "/var/cache/bind";
// ...
// forwarders {
// 158.109.0.9;
```

```
// 158.109.0.1;
// };
// query range you permit
    allow-query { localhost; 192.168.0.0/24; };
// the range to transfer zone files
    allow-transfer { localhost; 192.168.0.0/24; };
// recursion range you allow
// allow-recursion { localhost; 192.168.0.0/24; };


    dnssec-validation auto;
    auth-nxdomain no; # conform to RFC1035
//listen-on-v6 { any; };
};
```

Ara és necessari definir els arxius que realment resoldran les IP/nom de les zones. Comencem per *remix.server.lan*:

```
$TTL 86400
@ IN SOA nteum.remix.world. root.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

    IN NS nteum.remix.world.
    IN A 192.168.0.30
    IN MX 10 nteum.remix.world.

nteum IN A 192.168.0.30
```

S'ha de tenir en compte el "." al final dels noms de domini. El significat dels registres indiquen el següent: **SOA** (*Start of Authority*) ha d'estar en tots els arxius de zona a l'inici, després de `TTL` (*Time To Live*), que indica el temps en segons que els recursos d'una zona seran vàlids, el símbol `@` significa l'origen del domini; **NS**, el servidor de noms per al domini, **MX** (*Mail eXchange*) indica on s'ha de dirigir el correu per a una determinada zona, i **A** és l'IP que s'ha d'assignar a un nom. Per a l'arxiu de resolució inversa *0.168.192.db*:

```
$TTL 86400
@ IN SOA nteum.remix.world. nteum.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

    IN NS nteum.remix.world.
    IN PTR remix.world.
    IN A 255.255.255.0
30 IN PTR nteum.remix.world.
```

On podem observar que els registres tenen un format `<últim-dígit-IP> IN <PTR FQDN-of-system>` on PTR (*Registre PoinTeR*) crea un apuntador cap al nom de l'IP que coincideix amb el registre. De la mateixa manera procedim per a la zona externa amb l'arxiu *remix.world.wan*:

```
$TTL 86400
@ IN SOA nteum.remix.world. root.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

    IN NS nteum.remix.world.
    IN A 172.16.0.82
    IN MX 10 nteum.remix.world.
nteum IN A 172.16.0.82
```

I la seva resolució inversa *80.0.16.172.db*:

```
$TTL 86400
@ IN SOA nteum.remix.world. nteum.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

    IN NS nteum.remix.world.
    IN PTR remix.world.
    IN A 255.255.255.248
82 IN PTR nteum.remix.world.
```

Ara només restaria canviar el fitxer */etc/resolv.conf* per a incloure `nameserver 192.168.0.30` i reiniciar el servidor `service bind9 restart`. S'ha de mirar */var/log/syslog* per a observar si hi ha errors en l'arrencada del servidor i corregir-los fins que el seu reinici estigui lliure d'aquests (els més comuns són els símbols utilitzats com a comentaris, els punts al final de domini o espais abans dels registres A-PTR). Després, podem provar amb la instrucció `dig nteum.remix.world` i tindrem una sortida com:

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» nteum.remix.world
...
;; QUESTION SECTION:
;nteum.remix.world. IN A
;; ANSWER SECTION:
nteum.remix.world. 86400 IN A 192.168.0.30
;; AUTHORITY SECTION:
remix.world. 86400 IN NS nteum.remix.world.
;; Query time: 0 msec
...
```

Si fem la petició inversa amb `dig -x 192.168.0.30`, tindrem:

```
...
;; QUESTION SECTION:
;30.0.168.192.in-addr.arpa. IN PTR
;; ANSWER SECTION:
30.0.168.192.in-addr.arpa. 86400 IN PTR nteum.remix.world.
;; AUTHORITY SECTION:
0.168.192.in-addr.arpa. 86400 IN NS nteum.remix.world.
;; ADDITIONAL SECTION:
nteum.remix.world. 86400 IN A 192.168.0.30
...
```

Un aspecte molt interessant és, per exemple, posar àlies, la qual cosa significa incloure un registre CNAME en *remix.world.lan* com `ftp IN CNAME nteum.remix.world`. Atès que és un servei essencial, és necessari moltes vegades disposar d'un servei secundari (*slave*), i bind9 permet crear molt fàcilment serveis d'aquest tipus a partir del servidor primari per transferència de les taules (http://www.server-world.info/en/note?os=debian_7.0&p=dns&f=5).

1.1.3. Altres DNS

MaraDNS és un servidor DNS que pot funcionar com a DNS recursiu (cau) o com a *authoritative name server* i que es caracteritza per la seva extremadament fàcil configuració. MaraDNS ha estat optimitzat per a un petit nombre de dominis de manera simple i eficient i que permet crear un domini propi i servir-lo sense grans esforços, encara que conté tots els elements per a configurar-lo de manera similar a bind9. En el seu disseny, s'ha posat especial cura en la seguretat i utilitza una biblioteca de gestió de cadenes de caràcters que resisteixen els principals atacs als DNS per desbordament (*buffer overflows*)[6]. El paquet MaraDNS inclou un DNS autoritzat (`maradns`), un servidor DNS recursiu amb possibilitats de cau i que es diu Deadwood. La decisió de posar un servidor autoritzat o recursiu dependrà de la funció que es busqui. Si simplement és per a obtenir altres llocs a Internet, s'haurà de configurar un servidor recursiu. En canvi, si es desitja registrar dominis i es té una xarxa d'ordinadors dins d'aquest domini, cal configurar un servidor DNS amb autoritat.

Com a proves d'aquest servei, configurarem diferents opcions de MaraDNS per a Debian. Si bé el paquet està inclòs en el repositori, és una versió que té problemes de seguretat (http://maradns.samiam.org/debian.html) i el seu autor recomana baixar el codi font i compilar-lo/instal·lar-lo. Per a això, cal obtenir el fitxer des de l'adreça http://maradns.samiam.org/download.html, descomprimir-lo (`tar xjvf maradns-x.x.xx.tar.bz2` on x.x.xx és la versió del programari descarregat) i dins del directori executar `./configure; make` i si no hi ha errors executar `make install`. Els servidors s'instal·laran en */usr/local/sbin* i altres arxius complementaris en */usr/local/bin*. Els arxius de configuració en */etc/mararc* i */etc/dwood3rc* i el directori de configuració */etc/maradns*.

En primer lloc, per a configurar un DNS recursiu i cau, s'ha de modificar l'arxiu */etc/dwood3rc* per a incloure:

```
bind_address="127.0.0.1"
chroot_dir = "/etc/maradns"
#upstream_servers = {}
#upstream_servers["."]="8.8.8.8, 8.8.4.4"
recursive_acl = "127.0.0.1/16" # Qui pot fer servir la cau

# Paràmetres de configuració del servidor
maxprocs = 8 # Maximum number of pending requests
handle_overload = 1 # Send SERVER FAIL when overloaded
maradns_uid = 99 # UID Deadwood runs as
maradns_gid = 99 # GID Deadwood runs as
maximum_cache_elements = 60000
```

```
# File cache (readable and writable by the maradns_uid/gid)
cache_file = "dw_cache"
```

També és possible que Deadwood pugui contactar amb altres DNS recursius abans que amb els *root DNS servers*. Per a això, s'ha de treure el comentari de les línies upstream_servers (en aquest cas, s'hi han posat els DNS públics de Google). Després, podrem executar `/usr/local/sbin/Deadwood`, o si es vol executar com a *daemon*, s'haurà d'utilitzar el programa `duende` inclòs en el paquet: `/usr/local/bin/duende /usr/local/sbin/Deadwood`.

Per a configurar un DNS autoritzat, hem de modificar l'arxiu */etc/mararc* amb els següents valors: cvs2, el domini que gestionarà aquest servidor i l'arxiu que tindrà la informació en aquest cas db.local, el directori on es trobaran els arxius de configuració (/etc/maradns) i l'IP d'on respondrà.

```
csv2 = {}
csv2["remix.world."] = "db.local"
ipv4_bind_addresses = "127.0.0.1"
chroot_dir = "/etc/maradns"
```

L'arxiu */etc/maradns/db.local* només tindrà una línia per cada registre que desitgem que transformi el DNS, per exemple,

```
nteum.remix.world. 192.168.0.30 ~
```

A continuació s'ha d'engegar el servidor (`/usr/local/sbin/maradns`), o si es desitja executar com a *daemon*, aleshores s'haurà d'utilitzar el programa `/usr/local/bin/duende /usr/local/sbin/maradns`. Una de les qüestions interessants és la següent: com podem donar servei de manera *authoritative* per als nostres dominis en una Intranet però, a més, ser recursius quan la petició sigui a dominis externs? És a dir, tenim uns pocs noms (FQDN) que pertanyen al nostre domini (per exemple, remix.world) sobre la nostre Intranet i desitgem al mateix temps assegurar la resolució de domini FQDN externs (per exemple, debian.org). La solució plantejada per l'autor (S. Trenholme*) passa per una configuració combinada de MaraDNS 2.x i Deadwood 3.x posant MaraDNS que escolti sobre localhost (127.0.0.1) mentre que Deadwood ho fa sobre l'IP del servidor (192.168.1.37, en el nostre cas). Els arxius de configuració */etc/mararc* i */etc/dwood3rc* seran:

***http://woodlane.webconquest.com/pipermail/list/2011-august/000928.html**

```
# Arxiu /etc/mararc
# Simplement, indicar que escolti sobre localhost i el domini que serà responsable
ipv4_bind_addresses = "127.0.0.1"
timestamp_type = 2
verbose_level = 3
hide_disclaimer = "YES"
csv2 = {}
csv2["remix.world."] = "db.local"
chroot_dir = "/etc/maradns"

#Archivo /etc/dwood3rc
# Indicar-li que a més del nostre domini on volem consultar la resta
root_servers = {}
root_servers["remix.world."] = "127.0.0.1"
```

```
root_servers[”.”]=”198.41.0.4, 192.228.79.201, 192.33.4.12, 199.7.91.13,”
root_servers[”.”]+=”192.203.230.10, 192.5.5.241, 192.112.36.4, 128.63.2.53, ”
root_servers[”.”]+=”192.36.148.17, 192.58.128.30, 193.0.14.129, 199.7.83.42, ”
root_servers[”.”]+=”202.12.27.33”
# On escoltarà les peticions = IP del servidor Deadwood
bind_address=”192.168.1.37”
# Habilitar la resposta d'IP privades
filter_rfc1918=0
# IP que podran fer peticions
recursive_acl = ”192.168.1.0/24”
# Cau de disc per a Deadwood
cache_file = ”dw_cache”
# Directori arrel
chroot_dir = ”/etc/maradns”
```

Finalment, en les nostres màquines posarem com a */etc/resolv.conf* l'IP del nostre servidor `nameserver 192.168.1.37` i engegarem tots dos servidors (després de verificar que funciona, es poden posar com a *daemons* i baixar el nivell de *log* amb `verbose_level = 1`.

Un altre servidor interessant que ve en moltes distribucions és **Dnsmasq**, que permet de manera simple configurar un servidor DNS (*forwarder*) i un servidor DHCP en el mateix paquet per a xarxes petites/mitjanes. És possible atendre peticions de noms de màquines que no estan en els DNS globals i integrar el servidor de DHCP per a permetre que màquines gestionades pel DHCP apareguin en el DNS amb noms ja configurats, i a més suporta gestió d'IP dinàmiques i estàtiques pel DHCP i permet l'accés BOOTP/TFTP per a màquines sense disc i que fan la seva inicialització per xarxa.

La forma de configurar ràpidament un DNS per a màquines de domini propi i que faci de *forwarder* per als dominis externs és `apt-get update; apt-get install dnsmasq`. Si el que es desitja és un simple DNS, ja està configurat tenint en compte que en */etc/resolv.conf* tindrem alguna cosa com `nameserver 8.8.8.8`. Amb això, podem provar els externs utilitzant la instrucció `dig debian.org @localhost` (o simplement `dig debian.org`), però també respondrà a totes les màquines que tinguem definides en */etc/hosts*. Per exemple, si tenim una línia com `192.168.1.37 nteum.remix.world nteum` podrem executar `dig nteum @localhost` i ens respondrà amb l'IP corresponent. Per aquest motiu, si la xarxa és simple podem agregar les màquines en aquest arxiu, però si és més complexa, podem utilitzar l'arxiu de configuració */etc/dnsmasq.conf* que inclou una gran quantitat d'opcions per a organitzar els dominis interns i altres paràmetres no només per a la configuració del DNS, sinó també per al servidor de DHCP.[7]

1.2. NIS (YP)

Amb la finalitat de facilitar l'administració i donar comoditat a l'usuari en xarxes de diferents mides que executen GNU/Linux (o algun altre sistema operatiu amb suport per a aquest servei), s'executen serveis de *Network Information Service*, NIS (o *Yellow Pages*, YP, en la definició original de Sun). GNU/Linux pot donar suport com a client/servidor de NIS. La informació

que es pot distribuir en NIS és aquesta: usuaris (*login names*), contrasenyes (*passwords*, `/etc/passwd`), directoris d'usuari (*home directories*) i informació de grups (*group information*, `/etc/group`), xarxes, *hosts*, etc. Això presenta l'avantatge que, des de qualsevol màquina client o des del mateix servidor, l'usuari es podrà connectar amb el mateix compte i contrasenya i en el mateix directori (encara que el directori haurà de ser muntat anteriorment sobre totes les màquines client per NFS o mitjançant el servei de `automount`). [10, 11]

L'arquitectura NIS és del tipus client-servidor, és a dir, hi ha un servidor que disposarà de totes les bases de dades i uns clients que consultaran aquestes dades de manera transparent per a l'usuari. Per això, s'ha de pensar en la possibilitat de configurar servidors "de reforç" (anomenats *secundaris*) perquè els usuaris no quedin bloquejats davant la caiguda del servidor principal. Per aquest motiu, l'arquitectura es denomina realment *de múltiples servidors* (*master+mirrors-clients*).

1.2.1. Com es pot iniciar un client local de NIS en Debian?

Un client local és el que annexa l'ordinador a un domini NIS ja existent. Primer s'ha de verificar que es tenen instal·lats els paquets `netbase` (xarxa bàsica TCP/IP) i `rpcbind` (servidor que converteix nombres RPC –*Remote Procedure Call*– en ports DARPA) i `nis` (específic). És important destacar que el paquet `rpcbind`, que ja està en la majoria de distribucions, és un servei que substitueix el tradicional `portmap`. Aquest ha quedat obsolet per diferents causes vinculades als canvis en el kernel de NFSV3/V4, però especialment, perquè no té suport per a IPv6. És necessari verificar la instal·lació dels dos primers paquets per a tots els programes que executen RPC, incloent-hi NFS i NIS. Es recomana fer servir l'ordre `apt-get` (o també `dpkg, aptitude, synaptic`), i es pot verificar si està instal·lat amb la instrucció `apt-cache pkgnames` en mode text.

El procediment per a la instal·lació del paquet NIS sol·licitarà un domini (`NIS domainname`). Aquest és un nom que descriurà el conjunt de màquines que utilitzaran el NIS (no és un nom de *host*). Cal tenir en compte que `NISNteum` és diferent de `Nisnteum` com a nom de domini. Per a configurar-lo, es pot utilitzar l'ordre `nisdomainname`, domini que s'emmagatzemarà en `/proc/sys/kernel/domainname`. També es pot reconfigugar el paquet amb `dpkg-reconfigure nis` i tornarem a començar, podem veure que el rpcbind està en marxa amb `ps -edaf | grep rpcbind` o també amb `rpcinfo -p` i ens mostrarà diferents línies (per a diferents programes, ports/protocols, versions) com a *portmapper*. Per a reiniciar el `rpcbind`, podem fer `service rpcbind restart`. És important tenir en compte que la primera vegada que instal·lem en NIS, que serà el client en el nostre cas, el sistema d'instal·lació intentarà posar els *daemons* en marxa però fallarà, ja que no tenim cap servidor configurat (que serà el següent pas) i donarà un missatge de *[....] Starting NIS services: ypbind[....] binding to YP server...failed (backgrounded)*.

El client NIS utilitza l'ordre `ypbind` per a trobar un servidor per al domini especificat, mitjançant `broadcast` (no aconsellat per insegur) o buscant el servidor indicat en l'arxiu de configuració `/etc/yp.conf` (recomanable), que pot tenir bàsicament dos tipus de configuració com les mostrades a sota –només s'ha d'indicar una d'aquestes, encara que és recomanable la primera.

Sintaxi de l'arxiu `/etc/yp.conf`

`domain nisdomain server hostname`: indica que s'utilitza el *hostname* per al domini `nisdomain`. En el nostre cas, podria ser `domain NISnteum server 192.168.1.30`. Es podria tenir més d'una entrada d'aquest tipus per a un únic domini.

`ypserver hostname`: indica que s'utilitza *hostname* com a servidor introduint l'adreça IP del servidor NIS. Si s'indica el nom, ens hem d'assegurar que es pot trobar la IP per DNS o que la mateixa figura en l'arxiu `/etc/hosts`, ja que, d'una altra manera, el client es bloquejarà.

Per a iniciar el servei, s'ha d'executar:

`/etc/init.d/nis stop` *per a aturar-lo*
`/etc/init.d/nis start` *per a iniciar-lo*
o també amb l'ordre `service nis start|stop`

A partir d'aquest moment, el client NIS estarà funcionant. Això es pot confirmar amb `rpcinfo -o localhost ypbind`, que mostrarà les dues versions del protocol actiu. Quan el servidor estigui funcionant, es pot utilitzar l'ordre `ypcat mapname` (per exemple, `ypcat passwd`, que mostrarà els usuaris NIS definits en el servidor), en què les relacions entre *mapnames* i les taules de la base de dades NIS estan definides en `/var/yp/nicknames`.

1.2.2. Quins recursos s'han d'especificar per a utilitzar el NIS?

A partir de la inclusió de lib6 en les distribucions de GNULinux (això és Debian5 o FC4), és necessari orientar la consulta del *login* a les bases de dades adequades amb els passos següents:

1) Verificar el fitxer `/etc/nsswitch.conf` i assegurar-se que les entrades `passwd`, `group`, `shadow` i `netgroup` són similars a:

```
passwd: compat nis
group: compat nis
shadow: compat nis
netgroup: nis
hosts: files dns nis
```

2) Consulteu la sintaxi d'aquest arxiu en `man nsswitch.conf`.

3) En determinades configuracions, pot ser necessari agregar al final del fitxer */etc/pam.d/common-session* (si no existeix la línia)

```
session optional pam_mkhomedir.so skel=/etc/skel umask=077
```

per a crear automàticament el directori *home* si no existeix, però no és l'habitual, ja que generalment es desitja que aquest es munti del servidor NFS.

4) Amb aquesta configuració, es podrà fer una connexió local (sobre el client NIS) a un usuari que no estigui definit en el fitxer `/etc/passwd`, és a dir, un usuari definit en una altra màquina (*ypserver*). Per exemple, es podria fer `ssh -l user localhost`, en què `user` és un usuari definit en *ypserver*.

1.2.3. Com s'ha de configurar un servidor?

Abans d'instal·lar el servidor, cal tenir instal·lat el paquet `nis` i `rpcbind` sobre la màquina que farà de servidor (`apt-get install nis rpcbind`) i configurar el domini -el mateix que hem introduït quan hem configurat el client (NISnteum en el nostre cas). Després, s'ha de modificar l'arxiu */etc/default/nis* per a modificar la línia `NISSERVER=master` per a indicar-li el rol de servidor. També (encara que es pot fer en un segon pas) s'ha d'ajustar l'arxiu */etc/ypserv.securenets* per a modificar la línia que indica `0.0.0.0 0.0.0.0` que dóna accés a tothom i restringir-lo a la nostra xarxa, p. ex. `255.255.255.0 192.168.1.0`. Finalment, s'ha de modificar el */etc/hosts* perquè tinguem la màquina amb el nom definit en */etc/hostname* (es pot canviar/visualitzar amb l'ordre `hostname` o editant el fitxer) amb una línia FQND com, per exemple, `192.168.1.30 nteum.remix.world nteum`. La configuració del servidor es porta a terme amb l'ordre `/usr/lib/yp/ypinit -m`; en algunes distribucions, cal verificar que existeix l'arxiu `/etc/networks`, que és imprescindible per a aquest *script*. Si aquest arxiu no existeix, n'heu de crear un de buit amb `touch/etc/networks`; aquest *script* ens sol·licitarà quins seran els servidors NIS indicant-nos la màquina local per defecte, i haurem d'acabar amb Crtl+D. També es pot executar el client `ypbind` sobre el servidor; així, tots els usuaris entren per NIS indicant en l'arxiu */etc/default/nis* que existeixi `NISCLIENT=true`.

Considereu que, a partir d'aquest moment, les ordres per a canviar la contrasenya o la informació dels usuaris, com `passwd`, `chfn` o `adduser`, no són vàlides. En el seu lloc, s'hauran d'utilitzar ordres com ara `yppasswd`, `ypchsh` i `ypchfn`. Si es canvien els usuaris o es modifiquen els arxius esmentats, caldrà reconstruir les taules de NIS executant l'ordre `make` en el directori `/var/yp` per a actualitzar les taules.

La configuració d'un servidor esclau és similar a la del mestre, excepte si `NISSERVER = slave` en `/etc/default/nis`. Sobre el mestre, s'ha d'indicar que distribueixi les taules automàticament als esclaus, posant `NOPUSH = "false"` en l'arxiu `/var/yp/Makefile`. Ara s'ha d'indicar al mestre qui és el seu esclau mitjançant l'execució de:

```
/usr/lib/yp/ypinit -m
```

i introduint els noms dels servidors esclaus. Això reconstruirà els mapes, però no enviarà els arxius als esclaus. Per a això, sobre l'esclau, executeu:

```
/etc/init.d/nis stop
/etc/init.d/nis start
/usr/lib/yp/ypinit -s nombre_master_server
```

D'aquesta manera, l'esclau carregarà les taules des del mestre. També es podria posar en el directori `/etc/cron.d` l'arxiu NIS amb un contingut similar a (recordeu fer un `chmod 755 /etc/cron.d/nis`):

```
20 *  * * *  root  /usr/lib/yp/ypxfr_1perhour >/dev/null 2>&1
40 6 * * *    root  /usr/lib/yp/ypxfr_1perday  >/dev/null 2>&1
55 6,18 * * *    root  /usr/lib/yp/ypxfr_2perday  >/dev/null 2>&1
```

Amb això, es garantirà que tots els canvis del mestre es transfereixin al servidor NIS esclau.

Inicieu el servidor executant primer l'ordre `/etc/init.d/nis stop` i després la instrucció `/etc/init.d/nis start`. Aquesta sentència iniciarà el servidor (`ypserv`) i el *password daemon* (`yppasswdd`), l'activació del qual es podrà consultar amb `ypwich -d domain`.

Actualització de les taules NIS

És recomanable que després de fer servir `adduser o useradd` per a agregar un nou usuari sobre el servidor, executeu `cd /var/yp; make` per a actualitzar les taules NIS (i sempre que es canviï alguna característica de l'usuari, per exemple la paraula clau amb l'ordre `passwd`, només canviarà la contrasenya local i no la del NIS).

Per a provar que el sistema està funcionant i que l'usuari donat d'alta està en el NIS, podeu fer `ypmatch userid passwd`, on `userid` és l'usuari donat d'alta amb `adduser` abans i després d'haver fet el `make`. Per a verificar el funcionament del sistema NIS, podeu utilitzar l'*script* de http://tldp.org/howto/nis-howto/verification.html que permet una verificació més detallada del NIS.

També es pot provar en la mateixa màquina que un usuari es pot connectar a un domini NIS fent: `adduser usuari` i responent les preguntes, després `cd /var/yp; make`. En aquest moment, ho veurem tant en */etc/passwd* com en NIS amb `ypcat passwd`. Després, ho podem esborrar amb l'ordre `userdel usuari` (no s'ha de fer servir l'ordre `deluser` sinó el `userdel`), que només ho esborrarà del */etc/passwd* i no del NIS (cal anar amb compte de no fer un `make`, ja que aquest també ho esborrarà del NIS). Després ens podem connectar com a `ssh usuario@localhost`, i amb això podrem verificar que el NIS funciona, ja que l'accés només es podrà fer per NIS perquè per a */etc/passwd* no existeix aquest usuari. En relació amb NIS+, el seu desenvolupament va ser aturat el 2012 a causa de la falta d'interès/recursos de la comunitat, no així NIS/YP, que continua actualitzat i present en totes les distribucions (http://www.linux-nis.org/nisplus/).

1.3. Serveis de connexió remota: Telnet i ssh

1.3.1. Telnet i telnetd

Telnet és una ordre (client) utilitzat per a comunicar-se de manera interactiva amb un altre *host* que executa el *daemon* `telnetd`. L'ordre `telnet` es pot executar com `telnet host` o interactivament com `telnet`, el qual posarà el *prompt* "telnet>" i després, per exemple, `open host`. Una vegada establerta la comunicació, s'haurà d'introduir l'usuari i la contrasenya sota la qual es desitja connectar al sistema remot. Es disposa de diverses ordres (en mode interactiva) com `open`, `logout` i `mode` (s'han de definir les característiques de visualització), `close`, `encrypt`, `quit`, `set` i `unset` o es poden executar ordres externes amb "!". Es pot utilitzar l'arxiu `/etc/telnetrc` per a definicions per defecte o `.telnetrc` per a definicions d'un usuari particular (haurà d'estar en el directori *home* de l'usuari).

El *daemon* `telnetd` és el servidor de protocol Telnet per a la connexió interactiva. Generalment, és el *daemon* `inetd` que engega `telned`; es recomana incloure un `wrapper tcpd` (que utilitza les regles d'accés en `host.allow` i `host.deny`) en la crida al `telnetd` dins de l'arxiu `/etc/inetd.conf`. Per a incrementar la seguretat del sistema s'hi hauria de incloure, per exemple, una línia com:

```
telnet stream tcp nowait telnetd.telenetd /usr/sbin/tcpd /usr/bin/in.telnetd
```

En les distribucions actuals, la funcionalitat de `inetd` s'ha reemplaçat per la de `xinetd`, que requereix la configuració de l'arxiu `/etc/xinetd.conf` que s'ha explicat en l'apartat de configuració de xarxa de l'assignatura *Administració GNU/Linux*. Si es vol engegar `inetd` a mode de proves, es pot fer servir la sentència `/etc/init.d/inetd.real start`. Si l'arxiu `/etc/uissue.net` està present, el `telnetd` mostrarà el seu contingut a l'inici de la sessió. També es pot usar `/etc/security/access.conf` per a habilitar i deshabilitar *logins* d'usuari, *host* o grups d'usuaris, segons es connectin.

S'ha de recordar que, si bé el parell `telnet`-`telnetd` pot funcionar en mode *encrypt* en les últimes versions (transferència de dades encriptades, que han d'estar compilades amb l'opció corresponent), és una ordre que ha quedat en l'oblit per la seva falta de seguretat (transmet el text net sobre la xarxa, la qual cosa permet la visualització del contingut de la comunicació entre dues màquines, per exemple, amb l'ordre `tcpdump`); no obstant això, es pot fer servir en xarxes segures o situacions controlades.

Si no està instal·lat, es pot utilitzar (Debian) `apt-get install telnetd` i després verificar que s'ha donat d'alta, o bé en `/etc/inetd.conf`, o bé en `/etc/xinetd.conf` (o en el directori on estiguin definits els arxius; per exemple, `/etc/xinetd.d` segons s'indiqui en l'arxiu anterior amb la sentència

include /etc/xinetd.d). L'arxiu xinetd.conf o l'arxiu /etc/xinetd.d/telnetd hauran d'incloure una secció com*:

*Qualsevol modificació en xinetd.conf haurà d'arrencar novament el servei amb service xinetd restart.

```
service telnet {
disable = no
flags = REUSE
socket_type = stream
wait = no
user = root
server = /usr/sbin/in.telnetd
log_on_failure += USERID
}
```

SSL telnet(d)

Es recomana que en lloc de fer servir telnetd s'utilitzi SSL telnet(d), que reemplaça a telnet(d) utilitzant encriptació i autenticació per SSL, o que s'utilitzi SSH (el qual ja és un estàndard en totes les distribucions i entorns). L'SSLTelnet(d) pot funcionar amb el telnet(d) normal en ambdues adreces ja que a l'inici de la comunicació verifica si l'altre costat (*peer*) suporta SSL, i en cas contrari continua amb el protocol telnet normal. Els avantatges pel que fa a SSL telnet(d) són que les seves contrasenyes i dades no circularan per la xarxa en mode de text net i ningú que utilitzi l'ordre abans esmentada (tcpdump) podrà veure el contingut de la comunicació. També SSL telnet es pot utilitzar per a connectar-se, per exemple, a un servidor web segur (per exemple https://servidor.web.org) simplement fent telnet servidor.web.org 443.

1.3.2. SSH, *Secure shell*

Un canvi aconsellable avui dia és utilitzar ssh (ordre que ja està en tots els sistemes operatius, també en Windows, amb putty* o moabterm**) en lloc de telnet, rlogin o rsh. Aquestes tres últimes ordres són insegures (excepte SSLTelnet) per diverses raons. La més important és que tot el que es transmet per la xarxa, inclosos noms d'usuaris i contrasenyes, és en text net (encara que hi ha versions de telnet-telnetd encriptats, ha de coincidir que tots dos ho siguin), de manera que qualsevol que tingui accés a aquesta xarxa o a algun segment d'aquesta pot obtenir tota aquesta informació i després suplantar la identitat de l'usuari. La segona és que aquests ports (telnet, rsh, etc.) són el primer lloc on un *cracker* intentarà connectar-se. El protocol ssh (en la seva versió OpenSSH) proporciona una connexió encriptada i comprimida molt més segura que, per exemple, telnet (és recomanable utilitzar la versió 2.0 o versions superiors del protocol). Totes les distribucions actuals incorporen el client ssh i el servidor sshd per defecte. És necessari tenir actualitzada la biblioteca OpenSSL, utilitzada per aquests programes, ja que recentment es va trobar un problema de seguretat anomenat *heartbleed* i que ha estat ràpidament solucionat en la majoria de distribucions GNU/Linux (http://www.kriptopolis.com/recomendaciones-heartbleed).

*http://www.putty.org
**http://mobaxterm.mobatek.net

ssh

Per a executar l'ordre, feu:

```
ssh -l login-name host o ssh user@hostname
```

Mitjançant SSH, es poden encapsular altres connexions com X11 o qualsevol altra TCP/IP. Si s'omet el paràmetre `-l`, l'usuari es connectarà amb el mateix usuari local i en tots dos casos el servidor sol·licitarà la contrasenya per a validar la identitat de l'usuari. SSH suporta diferents modes d'autenticació (vegeu `man ssh`) basades en algorisme RSA i clau pública. Si el client no està instal·lat, cal fer `apt-get install openssh-client`.

Usant l'ordre `ssh-keygen -t rsa|dsa`, es poden crear les claus d'identificació d'usuari. L'ordre crea en el directori del `.shh` de l'usuari els fitxers* `id_rsa` i `id_rsa.pub`, les claus privada i pública, respectivament. L'usuari podria copiar la clau pública (`id_rsa.pub`) en l'arxiu `$HOME/.ssh/authorized_keys` del directori `.ssh` de l'usuari de la màquina remota. Aquest arxiu podrà contenir tantes claus públiques com llocs des d'on es vulgui connectar a aquesta màquina de manera remota. La sintaxi és d'una clau per línia, encara que les línies tindran una grandària considerable. Després d'haver introduït les claus públiques de l'usuari-màquina en aquest arxiu, aquest usuari es podrà connectar sense contrasenya des d'aquesta màquina. També per a efectuar aquesta còpia, es pot utilitzar l'ordre `ssh-copy-id màquina`, que copiarà les claus de manera automàtica i és molt més simple d'utilitzar.

***Per exemple, per a l'algorisme d'encriptació RSA.**

De manera normal (si no s'han creat les claus), es demanarà a l'usuari una contrasenya, però com que la comunicació serà sempre encriptada, mai serà accessible a altres usuaris que puguin escoltar sobre la Xarxa. Per a major informació, consulteu `man ssh`. Per a executar remotament una ordre, simplement feu:

```
ssh -l login name host_ordre_remot
```
Per exemple: `ssh -l user localhost ls -al`

sshd

El `sshd` és el servidor (*daemon*) per al `ssh` (si no estan instal·lats, es pot fer amb `apt-get install openssh-client openssh-server`. Junts reemplacen `rlogin`, `telnet`, `rsh` i proveeixen una comunicació segura i encriptada entre dos *hosts* insegurs de la Xarxa. Aquest s'arrenca generalment mitjançant els arxius d'inicialització (`/etc/init.d` o `/etc/rc`) i espera connexions dels clients. El `sshd` de la majoria de distribucions actuals suporta les versions 1, 2 i 3 del protocol SSH. Quan s'instal·la el paquet, es crea una clau RSA específica del *host* i quan el *daemon* s'inicia, en crea una altra, l'RSA per a la sessió, que no s'emmagatzema en el disc i que canvia cada hora. Quan un client inicia la comunicació, genera un nombre aleatori de 256 bits que està encriptat amb les dues claus del servidor i és enviat. Aquest nombre s'utilitzarà durant la comunicació com a clau de sessió per a encriptar la comunicació que es farà per mitjà d'un algorisme d'encriptació estàndard. L'usuari pot seleccionar qualsevol dels algorismes disponibles oferts pel servidor. Hi ha algunes diferències (més

segur) quan s'utilitza la versió 2 (o 3) del protocol. A partir d'aquest moment, s'inicien alguns dels mètodes d'autenticació d'usuari descrits en el client o se li sol·licita la contrasenya, però sempre amb la comunicació encriptada. Per a més informació, consulteu `man sshd`.

Tant `ssh` com `sshd` tenen un gran conjunt de paràmetres que poden ser configurats segons es necessiti en */etc/ssh/ssh_config* i */etc/ssh/sshd_config*. En el servidor, probablement algunes de les opcions més usades són `PermitRootLogin yes|no` per a permetre la connexió de l'usuari *root* o no, `IgnoreRhosts yes` per a evitar llegir els arxius *$HOME/.rhosts* i *$HOME/.shosts* dels usuaris, *X11Forwarding yes* per a permetre que aplicacions Xwindows en el servidor es visualitzin a la pantalla del client (molt útil en l'administració remota de servidors amb l'ordre `ssh -X host_per_a_administrar`).

Túnel sobre SSH

Moltes vegades tenim accés a un servidor `sshd`, però per qüestions de seguretat no podem accedir a altres serveis que no estan encriptats (per exemple, un servei de consulta de correu POP3 o un servidor de finestres X11) o simplement es vol connectar amb un servei al qual només es té accés des de l'entorn de l'empresa. Per a això, és possible establir un túnel encriptat entre la màquina client (per exemple, amb Windows i un client ssh anomenat `putty` de programari lliure) i el servidor amb `sshd`. En aquest cas, en vincular el túnel amb el servei, el servei veurà la petició com si vingués de la mateixa màquina. Per exemple, si volem establir una connexió per a POP3 sobre el port 110 de la màquina remota (i que també té un servidor `sshd`), farem:

```
ssh -C -L 1100:localhost:110 usuario-id@host
```

Aquesta ordre demanarà la contrasenya per a l'usuari-id sobre *host*, i una vegada connectat, s'haurà creat el túnel. Cada paquet que s'enviï des de la màquina local sobre el port 1100 s'enviarà a la màquina remota *localhost* sobre el port 110, que és on escolta el servei POP3 (l'opció `-C` comprimeix el trànsit pel túnel).

Fer túnels sobre altres ports és molt fàcil. Per exemple, suposem que només tenim accés a un *remote proxy server* des d'una màquina remota (*remote login*), no des de la màquina local; en aquest cas, es pot fer un túnel per a connectar el navegador a la màquina local. Considerem que tenim *login* sobre una màquina passarel·la (*gateway*), que pot accedir a la màquina anomenada *proxy*, la qual executa l'*Squid Proxy Server* sobre el port 3128. Executem:

```
ssh -C -L 8080:proxy:3128 user@gateway
```

Després de connectar-nos, tindrem un túnel escoltant sobre el port local 8080 que reconduirà el trànsit des de *gateway* cap a *proxy* al 3128. Per a navegar de manera segura, només s'haurà de fer `http://localhost:8080/`.

1.3.3. VPN SSL (via *tun driver*)

OpenSSH v4.3 introdueix una nova característica que permet crear *on-the-fly* VPN via el túnel *driver* (*tun*) com un pont entre dues interfícies en diferents segments de xarxa a través d'Internet i per la qual cosa un ordinador (B en el nostre cas) "quedarà dins" de la xarxa de l'altre ordinador (A) malgrat estar dins d'una altra xarxa. Per a analitzar aquest mecanisme, considerarem que l'ordinador A i B estan connectats a la xarxa via Ethernet i a Internet a través d'una *gateway* que fa NAT. A té IP sobre la seva xarxa (privada) 10.0.0.100, i B sobre la seva xarxa (privada) 192.168.0.100 i, com vam dir, cadascun té accés a Internet NAT *gateway*.

A través d'OpenSSH, connectarem B a la xarxa d'A via una interfície túnel ssh. Com hem definit, A ja té IP sobre la xarxa A (10.0.0.100) i que es veu des del *gateway* extern com a adreça 1.2.3.4, és a dir aquesta és la IP pública del servidor `ssh` de A (l'encaminador haurà de fer un *forwarding* d'1.2.3.4:22 a 10.0.0.100:22 perquè externament es pugui contactar amb servidor `ssh` de A). B també ha de tenir una IP sobre la xarxa A que serà la seva interfície tun0 i a la qual assignarem 10.0.0.200. B també ha de tenir accés al servidor `ssh` de A (o bé accés directe, o el port 22 ha de ser *forwarded* sobre la xarxa A en l'adreça 1.2.3.4). Quan el túnel estigui creat, B accedirà directament a la xarxa, la qual cosa significa que B "estarà" dins de la xarxa de A.

Els passos de configuració són els següents:

1) Activar l'IP *forwarding*: `echo 1 > /proc/sys/net/ipv4/ip_forward`

2) Túnel. Sobre B `ssh -w 0:0 1.2.3.4` es pot utilitzar com a opcions, a més, `-NTCf`. Això crea un *tunnel interface* tun0 entre client (B) i el servidor (A), recordeu que A té adreça pública (1.2.3.4). S'haurà de tenir accés de *root* a tots dos sistemes perquè puguin crear les interfícies (en verificar A que es té en `sshd_config PermitRootLogin yes` i `PermitTunnel yes`). Es recomana utilitzar SSH *keys* (PKI), per la qual cosa s'haurà de canviar PermitRootLogin without-password i si sobre el client no es vol donar accés de *root*, es pot utilitzar l'ordre `sudo` (vegeu `man sudoers`).

3) Verificar les interfícies: `ip addr show tun0`

4) Configurar les interfícies:
Ordinador A: `ip link set tun0 up ip addr add 10.0.0.100/32 peer 10.0.0.200 dev tun0`
Ordinador B: `ip link set tun0 up ip addr add 10.0.0.200/32 peer 10.0.0.100 dev tun0`

5) Provar: sobre B `ping 10.0.0.100` i sobre A `ping 10.0.0.200`

6) *Routing* directe: tenim un enllaç que connecta B a la xarxa A tot i que és necessari inicialitzar el *routing*. Sobre B: `ip route add 10.0.0.0/24 via 10.0.0.200` (per a permetre enviar des de B a qualsevol màquina de la xarxa A).

7) *Routing invers*: també perquè els paquets tornin a B sobre A hem de fer `arp -sD 10.0.0.200 eth0`

El *routing* ens assegurarà que altres màquines que estan en la xarxa A puguin enviar paquets a B a través d'A. Podem provar fent des de B `ping 10.0.0.123`. Si es desitja que tots els paquets de B vagin a través de A, haurem de canviar el *default gateway* de B però això no és simple ja que el mateix túnel va per Internet. Primer hem de crear un *host-based route* cap a la màquina A (tots els paquets excepte els que creen el *link* han d'anar pel túnel, però òbviament no els que creen el túnel). Sobre B: `ip route add 1.2.3.4/32 via 192.168.0.1`. En aquest cas, 192.168.0.1 és el *default gateway* per a B, és a dir el *gateway* sobre la xarxa B que proveeix de connectivitat a Internet i que ens servirà per a mantenir el túnel. Després podem canviar el *default gateway* sobre B: `ip route replace default via 10.0.0.1`. Amb la qual cosa, tindrem que 192.168.0.1 és el *default gateway* de la xarxa B i 10.0.0.1 *default gateway* de la xarxa A. Ja que la màquina B està connectada a la xarxa d'A, estem indicant utilitzar la xarxa A com a *default gateway* en lloc de *default gateway* sobre la xarxa B (com seria habitual). Es pot verificar això amb l'ordre `tracepath google.com`.

1.3.4. Túnels encadenats

Moltes vegades, l'accés a xarxes internes només és possible a través d'un servidor SSH (que estan en la DMZ) i des d'aquest és possible la connexió cap a servidors SSH interns per a després arribar a la màquina SSH del nostre interès (i si volem copiar arxius o fer el *forwarding* de X11, pot ser tot un repte). La manera de fer-ho és primer connectar-nos a l'M1, després d'aquesta a l'M2 (a la qual només és possible connectar-se des d'M1) i des d'aquesta a l'M3 (a la qual només és possible connectar-se des d'M2). Una manera de facilitar l'autenticació és utilitzar l'*agent forwarding* amb el paràmetre -A i posant la nostra clau pública en tots els *$HOME/.ssh/authorized_keys*. Així, quan anem executant les diferents ordres, la petició de claus pot ser *forwarded* cap a la màquina anterior i així successivament. Les ordres seran `ssh -A m1.org` i des d'aquesta `ssh -a m2.org` i al seu torn `ssh -a m3.org`. Per a millorar això, podem automatitzar-ho amb `ssh -A -t m1.org ssh -A -t m2.org ssh -A m2.org` (s'ha inclòs l'opció -t perquè `ssh` hi assigni una pseudo-tty i només veurem la pantalla final). L'*applet* SSHmenu pot ajudar a automatitzar tot això (http://sshmenu.sourceforge.net/).

Una forma de gestionar efectivament aquesta connexió "multisalts" és mitjançant l'opció `ProxyCommand` de `ssh` (vegeu `man ssh_config`), que ens

permetrà connectar-nos de manera més eficient. Per a això, definirem en *$HOME/.ssh/config* les següents ordres:

```
Host m1
    HostName m1.org
Host m2
    ProxyCommand ssh -q m1 nc -q0 m2.org 22
Host m3
    ProxyCommand ssh -q m2 nc -q0 m3.org 22
```

On la primera ordre (quan fem `ssh m1`) ens connectarà a m1.org. Quan executem `ssh m2`, s'establirà una connexió ssh sobre m1 però l'ordre ProxyCommand utilitzarà l'ordre `nc` per a estendre la connexió sobre m2.org. El tercer és una ampliació de l'anterior, per la qual cosa quan executem `ssh m3` és com si executéssim una connexió a m1 i després ampliéssim aquesta a m2 i després a m3.[23]

1.4. Serveis de transferència de fitxers: FTP

L'FTP (*File Transfer Protocol*) és un protocol client/servidor (sota TCP) que permet la transferència d'arxius des d'un sistema remot i cap a un sistema remot. Un servidor FTP és un ordinador que executa el *daemon* `ftpd`.

Alguns llocs que permeten la connexió anònima sota l'usuari *anonymous* són generalment repositoris de programes. En un lloc privat, es necessitarà un usuari i una contrasenya per a accedir-hi. També és possible accedir a un servidor FTP mitjançant un navegador i generalment avui dia els repositoris de programes se substitueixen per servidors web (per exemple, Apache) o altres tecnologies com Bittorrent (que fa servir xarxes *peer to peer*, P2P) o servidors web amb mòduls de WebDav o l'ordre `scp` (*secure copy*) que forma part del paquet openssh-client o el parell `sftp/sftpd` que formen part dels paquets `openssh-client` i `openssh-server`, respectivament. No obstant això, es continua utilitzant en alguns casos i Debian, per exemple, permet l'accés amb usuari/contrasenya o la possibilitat de pujar arxius al servidor (si bé amb serveis web-dav també és possible fer-ho).

El protocol (i els servidors/clients que l'implementen) d'FTP per definició no és encriptat (les dades, usuaris i contrasenyes es transmeten en text net per la Xarxa), amb el risc que això representa. No obstant això, hi ha una sèrie de servidors/clients que suporten SSL i, per tant, encriptació.

1.4.1. Client FTP (convencional)

Un client FTP permet accedir a servidors FTP i hi ha una gran quantitat de clients disponibles. L'ús de l'FTP és summament simple; des de la línia d'ordres, executeu:

`ftp nom-servidor`

O també `ftp` i després, de manera interactiva: `open nom-servidor`

El servidor sol·licitarà un nom d'usuari i una contrasenya (si accepta usuaris anònims, s'introduirà *anonymous* com a usuari i la nostra adreça de correu electrònic com a contrasenya), i a partir del *prompt* de la línia d'ordres (després d'alguns missatges), podrem començar a transferir fitxers.

El protocol permet la transferència en mode ASCII o binari. És important decidir el tipus de fitxer que cal transferir perquè una transferència d'un binari en mode ASCII inutilitzarà el fitxer. Per a canviar d'un mode a un altre, s'han d'executar les ordres `ascii` o `binary`. Les ordres útils del client FTP són el `ls` (navegació en el directori remot), `get nom_del_fitxer` (per a descarregar fitxers) o `mget` (que admet `*`), `put nom_del_fitxer` (per a enviar fitxers al servidor) o `mput` (que admet `*`); en aquests dos últims, s'ha de tenir permís d'escriptura sobre el directori del servidor. Es poden executar ordres locals si abans de l'ordre s'insereix un "`!`". Per exemple, `!cd /tmp` significarà que els arxius que baixin a la màquina local es descarregaran en `/tmp`. Per a poder veure l'estat i el funcionament de la transferència, el client pot imprimir marques, o *ticks*, que s'activen amb les ordres `hash` i `tick`. Hi ha altres ordres que es poden consultar en el full del manual (`man ftp`) o fent `help` dins del client. Tenim nombroses alternatives per als clients, per exemple en mode text: `ncftp`, `lukemftp`, `lftp`, `cftp`, `yafc Yafc` o, en manera gràfica: `gFTP`, `WXftp`, `LLNL XFTP`, `guiftp`.

Enllaç d'interès

En la Viquipèdia disposeu d'una comparativa de diversos clients FTP: http://en.wikipedia.org/wiki/Comparison_of_FTP_client_programari

1.4.2. Servidors FTP

El servidor tradicional d'UNIX s'executa a través del port 21 i és engegat pel *daemon* `inetd` (o `xinetd`, segons es tingui instal·lat). En `inetd.conf` convé incloure el *wrapper* `tcpd` amb les regles d'accés en `host.allow` i el `host.deny` en la crida al `ftpd` per l'`inetd` per a incrementar la seguretat del sistema. Quan rep una connexió, verifica l'usuari i la contrasenya i el deixa entrar si l'autenticació és correcta. Un FTP *anonymous* treballa de manera diferent, ja que l'usuari només podrà accedir a un directori definit en l'arxiu de configuració i a l'arbre subjacent, però no cap amunt, per motius de seguretat. Aquest directori generalment conté directoris `pub/`, `bin/`, `etc/` i `lib/` perquè el *daemon* d'FTP pugui executar ordres externes per a peticions de `ls`. El *daemon* `ftpd` suporta els següents arxius per a la seva configuració:

Vegeu també

El tema de la seguretat del sistema s'estudia en el mòdul "Administració de seguretat".

- **/etc/ftpusers**: llista d'usuaris que no són acceptats en el sistema. Un usuari per línia.
- **/etc/ftpchroot**: llista d'usuaris als quals se'ls substituirà el directori base `chroot` quan es connectin. Necessari quan desitgem configurar un servidor anònim.

- **/etc/ftpwelcome**: anunci de benvinguda.
- **/etc/motd**: notícies després del *login*.
- **/etc/nologin**: missatge que es mostra després de negar la connexió.
- **/var/log/ftpd**: *log* de les transferències.

Si en algun moment volem inhibir la connexió a l'FTP, es pot incloure l'arxiu `/etc/nologin`. L'`ftpd` mostra el seu contingut i acaba. Si existeix un arxiu `.message` en un directori, l'`ftpd` el mostrarà quan s'hi accedeixi. La connexió d'un usuari passa per cinc nivells diferents:

1) tenir una contrasenya vàlida;

2) no aparèixer en la llista de `/etc/ftpusers`;

3) tenir un *shell* estàndard vàlid;

4) si apareix en `/etc/ftpchroot`, es canviarà al directori `home` (també si és *anonymous* o FTP);

5) si l'usuari és *anonymous* o FTP, llavors haurà de tenir una entrada en el `/etc/passwd` amb usuari FTP, però podrà connectar-se especificant qualsevol contrasenya (per convenció, s'utilitza l'adreça de correu electrònic).

És important prestar atenció al fet que els usuaris habilitats únicament per a utilitzar el servei FTP no disposin d'un *shell* a l'entrada corresponent d'aquest usuari en `/etc/passwd` per a impedir que aquest usuari tingui connexió, per exemple, per `ssh` o `telnet`. Per a això, quan es creï l'usuari, caldrà indicar, per exemple:

```
useradd -d/home/nteum -s /bin/false nteum
```
i després: `passwd nteum`,

la qual cosa indicarà que l'usuari nteum no tindrà *shell* per a una connexió interactiva (si l'usuari ja existeix, es pot editar el fitxer `/etc/passwd` i canviar l'últim camp per `/bin/false`). A continuació, s'ha d'agregar com a última línia `/bin/false` en `/ect/shells`. En [28] es descriu pas a pas com es crea tant un servidor FTP segur amb usuaris registrats com un servidor FTP *anonymous* per a usuaris no registrats. Dos dels servidors no estàndards més comuns són el ProFTPD i Vsftpd (tots dos inclosos en Debian, per exemple).

Per a fer la instal·lació del Proftpd sobre Debian, executeu `apt-get install proftpd`. Una vegada descarregat `debconf`, preguntarà si es vol executar per `inetd` o manualment (és recomanable triar l'última opció). Si s'ha d'aturar el servei (per a canviar la configuració, per exemple): `/etc/init.d/proftpd stop`. I per a modificar el fitxer: `/etc/proftpd.conf`. Un servidor (Debian)

Enllaços d'interès

Per a saber més sobre ProFTPD i Vsftpd, podeu visitar les següents pàgines: http://www.proftpd.org https://security.appspot.com /vsftpd.html.

Enllaços d'interès

Per a configurar un servidor FTP en mode encriptat (TSL) o per a tenir accés *anonymous*, podeu consultar: http://www.debian -administration.org /articles/228.
D'altra banda, per a saber més sobre la instal·lació i configuració de PureFtpd podeu consultar: http://www.debian -administration.org /articles/383.

molt interessant és el PureFtpd (`pure-ftpd`), que és molt segur, permet usuaris virtuals, quotes, SSL/TSL i té un conjunt de característiques interessants. El paquet virtual de Debian ftp-server (https://packages.debian.org/wheezy/ftp-server) mostra tots els paquets d'aquest tipus inclosos en l'última versió.

1.5. *Active Directory Domain Controller* amb Samba4

Un dels aspectes més importants en la integració de sistemes és la gestió d'usuaris i identificació centralitzada, així com les autoritats d'autenticació i els permisos. En molt llocs basats en Windows, aquesta tasca és portada a terme per un *Active Directory* (AD, servei de directori en una xarxa distribuïda d'ordinadors (de vegades, també anomenat PDC per *Primary Domain Controller*), i és important disposar d'eines de la banda d'*Open Source* que permetin gestionar aquest tipus d'entorns. Samba versió 4 és una nova distribució d'aquest popular programari que permet la integració amb sistemes Windows actuant com a servidor d'arxius i/o impressores i a més, en aquesta última versió i de manera estable, pot actuar com a controlador d'un domini Windows acceptant clients, gestionant usuaris i directoris de manera centralitzada i totalment compatible.[29, 30]

Dels experts d'*Active Directory* (que generalment són consultors especialitzats i amb un alt cost), sabem que AD és una unió (que pot ser molt complicada d'entendre per als administradors que no estiguin dedicats al món Windows) de diferents serveis i tecnologies com ara DNS, Kerberos, LDAP i CIFS. Alguns hi inclouen DHCP també, però com veurem no és necessari en la nostra instal·lació. Estaríem en un error si penséssim que configurant cadascun d'aquests serveis tindríem el resultat del conjunt, però Samba4 presenta una capa d'abstracció, estable i eficient, que els integra per a oferir un producte complex però que es pot configurar i administrar seguint un conjunt de passos sense grans dificultats (encara que no s'ha de pensar que és simple). L'element crític (base dels majors problemes i errors) de l'AD és el DNS (en realitat, Windows utilitzarà l'AD com a DNS), ja que aquest el farà servir per a "agregar extraoficialment" a la llista de noms habituals en un DNS els servidors AD perquè els clients puguin trobar-los i tenir-hi accés.

En relació amb Kerberos i LDAP, els administradors de GNU/Linux saben de la seva potencialitat i complexitat, però en el cas d'AD estan integrats en el paquet i si bé els atorga una certa estandardització del sistema no són integrables amb servidors o altres clients, només s'utilitzen per als seus objectius i particularment quan fem servir Samba4, serà aquest qui els configurarà i gestionarà en el nostre nom amb petites modificacions per part de l'administrador. La versió actual de Samba (V4) no difereix de les anteriors quant a compartició d'arxius i impressores (fins i tot, s'ha simplificat la seva gestió/administració), i a més, amb la implementació d'AD permetrà que aquells administradors que utilitzin Windows puguin continuar utilitzant les seves eines de gestió i administració de domini només apuntant al servidor Samba.

Tota la configuració de Samba passarà ara per l'arxiu `smb.conf` (normalment en */etc/samba/smb.conf*) amb definicions simplificades però permetent la gestió complexa d'un domini Windows mitjançant les eines (també complexes) de Windows com per exemple RSAT (*Remote Server Administration Tools*) i, òbviament, a través del sistema GNU/Linux i la CLI de Samba4 es tindrà accés a tota la configuració i administració de l'AD (sense necessitat d'utilitzar Windows en cap cas).[30, 33]

En la nostra instal·lació habitual, farem servir Debian Wheezy com a distribució, i si bé Samba4 està en els repositoris *backport* (en els de Wheezy només hi ha la versió 3.x però estarà disponible en la nova edició de Debian Jessie) optem, per la facilitat a l'hora de resoldre les dependències, per baixar el codi font i compilar-lo.

1.5.1. Passos preliminars

Una de les primeres verificacions que hem de fer és que NPT estigui instal·lat i funcionant correctament (`apt-get install ntp` i provar que se sincronitza correctament amb `ntpq -p`), ja que Kerberos és molt exigent quant als valors del rellotge. Un segon aspecte que cal cuidar és definir els servidors amb IP estàtiques (almenys durant la prova de concepte de la instal·lació i configuració de l'AD i, òbviament, en un servidor en producció). Per a això, definirem */etc/network/interfícies* amb:

```
auto eth0
iface eth0 inet static
    address 192.168.1.33
    netmask 255.255.255.0
    gateway 192.168.1.1
```

Després, haurem de verificar el */etc/hosts* (si és que no disposem d'un DNS on puguem modificar els registres A i PTR per a incloure el nostre servidor d'AD) per a agregar (en el nostre cas) una entrada com aquesta: `192.168.1.33 sysdw.nteum.org sysdw`. Com DNS, es podria fer servir qualsevol d'aquells que s'han vist anteriorment, però en aquest cas ens va semblar més adequat, per a fer una prova total de Samba4, utilitzar el que porta intern el paquet i que l'utilitzarà l'AD. Per aquest motiu, haurem d'estar segurs de desactivar qualsevol altre servei de DNS que tinguem en aquesta màquina perquè no hi hagi conflictes (mireu per exemple amb `netstat -anotup`). L'altre aspecte important és el nom de domini, i sense entrar en polèmiques en relació amb com es tracta en GNU/Linux i com es "veuen" en Windows, hem escollit per a simplificar el del DNS (si bé alguns experts indiquen que pot haver-hi conflictes quan tractem amb un DNS extern). És a dir, un *AD Domain* no significa realment un *DNS domain* sinó que és una "unió" híbrida de DNS i que Kerberos denomina `realm`. És per això que configurarem el */etc/resolv.conf* apuntant com a DNS a la mateixa màquina de Samba4 i amb el domini DNS d'aquesta (i com a secundari un de públic). Recordeu que si es té instal·lat i actiu el

paquet `resolvconf` les modificacions de */etc/resolv.conf* són dinàmiques i les inclusions dels DNS s'hauran de fer sobre */etc/network/interfícies* amb els *tags* `dns-nameservers` i `dns-search`:

```
domain nteum.org
search nteum.org
nameserver 192.168.1.33
nameserver 8.8.8.8
```

A més a més, cal verificar que */etc/hostname* coincideix amb el nom indicat en */etc/hosts*. Finalment, com que Samba4 actuarà com a servidor d'arxius i com a PDC/AD, és necessari assegurar que el sistema d'arxius que estiguem exportant suporti ACL (*access control list*) i atributs estesos, i per a això modificarem */etc/fstab* per a indicar al *file system* corresponent els atributs `user_xattr,acl`, incloent-hi una línia com:

```
UUID=.... / ext4 user_xattr,acl,errors=remount-ro 0 2.
```

Aquí ho hem fet sobre el directori *root* ja que tenim una partició només, però això només s'ha d'aplicar a aquelles particions que es desitgi servir als clients.

Finalment, reiniciarem la màquina perquè totes les configuracions s'actualitzin correctament.

1.5.2. Compilar Samba4

En primer lloc, hem de tenir una sèrie de paquets instal·lats (si bé molts d'aquests ja es troben instal·lats; aquesta llista pot variar segons les fonts que es consultin), però els requeriments de Samba4 indiquen:

```
apt-get install build-essential libacl1-dev libattr1-dev \
   libblkid-dev libgnutls-dev libreadline-dev python-dev libpam0g-dev \
   python-dnspython gdb pkg-config libpopt-dev libldap2-dev \
   dnsutils libbsd-dev attr krb5-user docbook-xsl libcups2-dev acl
```

Altres experts en Samba4 i AD també inclouen (algunes d'aquestes ja estaran incloses o es resoldran per dependències):

```
apt-get install  pkg-config libacl1 libblkid1 attr libattr1 libgnutls-dev  \
libncurses-dev libdm0-dev libfam0 fam libfam-dev xsltproc libnss3-dev \
docbook-xsl-doc-html docbook-xsl-ns python-dnspython libcups2-dev krb5-config
```

En el nostre cas, utilitzarem com a servidor de Kerberos el nostre servidor AD, per la qual cosa és important la seva configuració (paquets `krb5-user` i `krb5-config`) i haurem d'introduir durant la seva instal·lació el domini (*realm*) que servirà en lletres majúscules, que en el nostre cas serà NTEUM.ORG, i el *password* haurà de tenir certa complexitat (com per exemple PaSSw0d2014) perquè el programa el consideri vàlid. Per a obtenir i compilar l'última versió de Samba4, farem:

`wget http://www.samba.org/samba/ftp/stable/samba-4.x.y.tar.gz` (s'ha de reemplaçar x.y amb l'última versió).

```
tar -xzf samba-4.x.y.tar.gz
cd samba-4.x.y
./configure --prefix=/opt/samba4
make
make install
```

Nosaltres hem escollit com a lloc d'instal·lació */opt/samba4* , per la qual cosa haurem d'actualitzar la variable PATH amb (s'ha de posar en *.profile*):

```
export PATH=/usr/local/samba/bin:/usr/local/samba/sbin:$PATH
```

i perquè la configuració quedi com és habitual, es crea un enllaç a */etc/samba* amb `ln -s /usr/local/samba/etc /etc/samba`.[32, 33]

El següent és aprovisionar l'*AD Domain*:

```
samba-tool domain provision --use-rfc2307 --interactive
```

Aquí haurem d'indicar el *realm* (NTEUM.ORG, que és el mateix que configurem en krb5) i el domini (NTEUM en el nostre cas). És important que el *password* coincideixi amb el que hem introduït en krb5 (i tingui certa complexitat), i assegurar-se que el *DNS forwarder* apunti el *DNS server* real que resoldrà les peticions de DNS i introduir la línia `allow dns updates = nonsecure` en els paràmetres globals. L'arxiu */etc/samba/smb.conf* quedarà com:

```
[global]
workgroup = NTEUM
realm = nteum.org
netbios name = SYSDW
server role = active directory domain controller
dns forwarder = 158.109.0.9
allow dns updates = nonsecure

[netlogon]
path = /opt/samba4/var/locks/sysvol/nteum.org/scripts
read only = No

[sysvol]
path = /opt/samba4/var/locks/sysvol
read only = No
```

A continuació, hem d'ajustar la configuració de Kerberos fent:

```
mv /etc/krb5.conf /etc/krb5.conf.org
cp  /opt/samba4/private/krb5.conf .
```

El contingut de la qual haurà de quedar com:

```
[libdefaults]
default_realm = NTEUM.ORG
dns_lookup_realm = false
dns_lookup_kdc = true
```

Una vegada reiniciat el sistema, podem engegar el servidor simplement executant `samba` i mirar el */var/log/syslog* per a verificar que s'ha arrencat correctament, i si hi ha errors, de quin tipus són per a solucionar-los (si executem `ps -edaf | grep samba` veurem aproximadament 14 processos samba si tot ha anat bé). Podrem verificar l'execució del servidor fent:

```
smbclient -L localhost -U%
    Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
       Sharename       Type      Comment
   ---------       ----      -------
   netlogon        Disk
   sysvol          Disk
   IPC$            IPC       IPC Service
    Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
   Server               Comment
   ---------            -------
   Workgroup            Master
       ---------            -------

smbclient //localhost/netlogon -UAdministrator -c 'ls'      Ens sol·licitarà el passwd d'administrator
    Enter Administrator's password:
    Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
    .                                   D        0  Wed Jul 23 10:11:33 2014
    ..                                  D        0  Wed Jul 23 10:11:39 2014

44541 blocks of size 262144. 7486 blocks available
```

Per a verificar el DNS:

```
host -t SRV _ldap._tcp.nteum.org
    _ldap._tcp.nteum.org has SRV record 0 100 389 sysdw.nteum.org.

host -t SRV _kerberos._udp.nteum.org.
    _kerberos._udp.nteum.org has SRV record 0 100 88 sysdw.nteum.org.

host -t A sysdw.nteum.org
sysdw.nteum.org has address 158.109.65.67
```

També cal verificar que tenim connexió externa amb un `ping google.com`, per exemple.

Finalment, per a verificar el funcionament de KRB5 (el *realm* ha d'anar en majúscules):

```
kinit administrator@NTEUM.ORG
    Password for administrator@NTEUM.ORG:
    Warning: Your password will expire in 41 days on Wed Sep  3 17:10:52 2014
```

I podríem treure (si bé en entorns de producció no seria recomanable) la caducitat del *password* amb: `samba-tool user setexpiry administrator -noexpiry`

Finalment, si tot funciona haurem de repetir el procediment de recrear el domini però agregant `-use-ntvfs` com a paràmetre, per a això podem fer:

```
mv /etc/samba/smb.conf /etc/samba/smb.conf.org
samba-tool domain provision --realm nteum.org --domain NTEUM --adminpass PaSSw0d2014 \
   --server-role=dc --use-ntvfs
```

El nou arxiu *smb.conf* quedarà (agregant novament la línia `allow dns updates`):

```
[global]
workgroup = NTEUM
realm = nteum.org
netbios name = SYSDW
server role = active directory domain controller
dns forwarder = 158.109.0.9
server services = rpc, nbt, wrepl, ldap, cldap, kdc, drepl, winbind, ntp
_signd, kcc, dnsupdate, dns, smb
dcerpc endpoint servers = epmapper, wkssvc, rpcecho, samr, netlogon, lsa
rpc, spoolss, drsuapi, dssetup, unixinfo, browser, eventlog6, backupkey, dnsserv
er, winreg, srvsvc
allow dns updates = nonsecure
[netlogon]
path = /opt/samba4/var/locks/sysvol/nteum.org/scripts
read only = No

[sysvol]
path = /opt/samba4/var/locks/sysvol
read only = No
```

I que /etc/krb5.conf és:

```
[libdefaults]
default_realm = NTEUM.ORG
dns_lookup_realm = false
dns_lookup_kdc = true
```

S'ha de reiniciar novament i executar `samba` (recordeu que encara no el tenim com a servei i que l'hem d'executar manualment). Si volem que Samba4 serveixi els directoris *homes*, haurem d'agregar en *smb.conf* una secció amb:

```
[home]
        path = /home/
        read only = no
```

Perquè l'administrador tingui els privilegis correctes en la gestió dels comptes des d'entorns Windows (i que Samba4 creï els directoris *home* automàticament), haurem d'executar:

```
net rpc rights grant 'NTEUM\Domain Admins' SeDiskOperatorPrivilege
-Uadministrator.
```

Ara hauríem de comprovar que podem configurar un client Windows i agregar-lo al directori. Per a això (en el nostre cas), utilitzarem una màquina Windows8 on en primer lloc modificarem la interfície de xarxa perquè el DNS faci referència al nostre servidor d'AD (anem a *Control Panel* > *Network and Internet*

> *Network and Sharing Center*, seleccionem el dispositiu de xarxa actiu i a propietats hi canviem el DNS, i deixem només l'IP del nostre AD). Després anem a *Control Panel* > *System and Security* > *System* i seleccionem *Advanced System Settings* i dins *Computer Name* > *Change* i introduïm *administrator* i PaSSw0d2014 per a agregar la màquina al domini (en cas que no ens seleccioni el domini per defecte, podem posar a *Username*: NTEUM\administrator, i després el *passwd*), que ens donarà un missatge de confirmació si tot ha anat bé i ens demanarà fer un *reboot* de la màquina. Quan s'iniciï novament, podrem seleccionar un altre usuari/domini i connectar-nos a NTEUM\administrator (que és el compte prèviament definit). Amb tot això, es generaran una sèrie de passos en el client i el servidor per a aprovisionar en aquesta nova màquina la configuració i directori arrel.[32, 33]

Per a administrar el lloc AD, es poden utilitzar les eines de Windows (RSAT), les quals es poden obtenir (sense càrrec) des del lloc del fabricant i amb la guia indicada en [31] i exemples de configuració en [32]. També és possible utilitzar l'eina Swat (https://wiki.samba.org/index.php/swat2, última versió de novembre del 2012), però la seva instal·lació pot presentar alguns inconvenients (sobretot si Samba4 s'ha instal·lat des de les fonts). Finalment, en l'adreça https://wiki.samba.org/index.php/samba4/InitScript podrem trobar l'*script* per a iniciar i apagar el servidor AD de manera automàtica.

1.6. Serveis d'intercanvi d'informació en nivell d'usuari

1.6.1. El *Mail Transport Agent* (MTA)

Un MTA (*Mail Transport Agent*) s'encarrega d'enviar i rebre els correus des d'un servidor de correu electrònic cap a Internet i des d'Internet, que implementa el protocol SMTP (*Simple Mail Transfer Protocol*). Totes les distribucions incorporen diferents MTA i, per exemple, els de Debian es poden consultar en el seu paquet virtual *mail-transport-agent* (https://packages.debian.org/wheezy/mail-transport-agent). Una de les que s'utilitzen habitualment és `exim`, ja que és més fàcil de configurar que altres paquets MTA, com són `postfix` o `sendmail` (aquest últim és un dels precursors). `Exim` presenta característiques avançades com rebutjar connexions de llocs de *spam* coneguts, posseeix defenses contra correus brossa (*junk mails*) o bombardeig de correu (*mail bombing*), i és extremadament eficient en el processament de grans quantitats de correus.

La seva instal·lació és mitjançant `apt-get install exim4-daemon-heavy` (en aquest cas, s'optarà per la instal·lació de la versió *heavy*, que és la més completa i suporta llista d'accés, ACL, i característiques avançades, i en instal·lacions més senzilles es pot optar per *exim4-daemon-light*). La seva configuració es fa mitjançant `dpkg-reconfigure exim4-config`, on una resposta típica a les preguntes efectuades és:

- *General type of mail configuration*: *Internet site*; el correu és enviat i rebut utilitzat per SMTP.

- *System mail name*: remix.world (el nostre domini)
- *IP-addresses to listen on for incomming SMTP connections*: (deixar en blanc)
- *Other destinations for which mail is accepted*: remix.world
- *Domains to relay mail for*: (deixar en blanc)
- *Machines to relay mail for*: (deixar en blanc)
- *Keep number of DNS-queries minimal (Dial-on-Demand)*?: No
- *Delivery method for local mail: Maildir format in home directory*
- *Split configuration into small files*?: No

El servidor ja estarà configurat i es pot provar amb `echo test message | mail -s "test" adminp@SySDW.nteum.org` (canviant l'adreça, per descomptat) i verificar que el correu ha arribat a l'usuari adminp (els errors es poden trobar en */var/log/exim4/mainlog*). La configuració serà emmagatzemada en */etc/exim4/update-exim4.conf.conf*. Per a configurar autenticació per TLS, ACL i Spam Scaning, podeu consultar https://wiki.debian.org/exim.

1.6.2. External SMTP

Quan instal·lem un nou sistema com a servidors o estacions de treball, un aspecte rellevant és el servidor de correu i podem instal·lar grans paquets com els ja esmentats Postfix, Exim o Zimbra (en la seva versió Community http://www.zimbra.com/), fent que els correus cap a dominis externs utilitzin els serveis externs d'SMTP (per exemple, els de Google). Per a màquines virtuals, estacions de treball o portàtils és una mica més complicat ja que generalment tenen IP privades o en xarxes internes, per la qual cosa és necessari tenir un servidor que faci de receptor dels correus externs al nostre domini, és a dir, un servidor que faci les funcions de *smarthost*, per exemple el Google Apps SMTP. Per a detalls de la seva configuració, es pot seguir la documentació d'https://wiki.debian.org/gmailandexim4. D'acord amb la informació de Google (https://support.google.com/a/answer/2956491?hl=es) i per a comptes gratuïts, el nombre màxim de destinataris permès per domini i dia és de 100 missatges i Gmail reescriurà l'adreça del remitent. Per a la seva configuració, executarem `dpkg-reconfigure exim4-config` i seleccionarem:

- *mail sent by smarthost; received via SMTP or fetchmail*.
- *System mail name*: *localhost*
- *IP-addresses to listen on for incomming SMTP connections*: 127.0.0.1
- *Other destinations for which mail is accepted*: (deixar en blanc)
- *Machines to relay mail for*: (deixar en blanc)
- *IP address or host name of the outgoing smarthost*: smtp.gmail.com::587
- *Hide local mail name in outgoing mail*?: No
- *Keep number of DNS-queries minimal (Dial-on-Demand)*?: No
- *Delivery method for local mail: mbox format in /var/mail*
- *Split configuration into small files*?: Yes

Aquesta és la configuració més adequada si no es té un IP visible externament. L'enviament sobre el port 587 de Gmail utilitza STARTTLS per a assegurar la protecció del passwd i per a indicar l'usuari i passwd d'accés a Gmail (utilitzeu un compte només per a aquest objectiu, no el compte habitual de Gmail), s'ha d'editar el fitxer */etc/exim4/passwd.client* i agregar la següent línia.

```
*.google.com:SMTPAccountName@gmail.com:y0uRpaSsw0RD
```

Després, executar (per a evitar que altres usuaris de la màquina puguin llegir el seu contingut):

```
chown root:Debian-exim /etc/exim4/passwd.client
chmod 640 /etc/exim4/passwd.client
```

Gmail reescriurà l'adreça del remitent automàticament, però si no ho fes, o enviem un *smarthost* que no ho fa, hauríem de configurar l'arxiu */etc/email-addresses* amb totes les combinacions d'adreces possibles per a utilitzar (una per línia) i l'adreça que es reescriurà (per exemple, nteum@remix.world: Smtpaccountname@gmail.com). Després, s'haurà d'executar:

```
update-exim4.conf
invoke-rc.d exim4 restart
exim4 -qff
```

Amb això s'actualitza i recarrega la configuració, i es força a enviar tots els correus que estan pendents. Com vam mostrar amb anterioritat, en el fitxer */var/log/exim4/mainlog* tindrem els errors si n'hi ha. Si existeixen errors d'autenticació sobre Gmail, hem de verificar amb `host smtp.gmail.com` quins són els *hosts* que retorna i si aquests concorden amb el patró inclòs en */etc/exim4/passwd.client*. Si és diferent, canvieu-lo perquè coincideixi.

1.7. *Internet Message Access Protocol* (IMAP)

Aquest servei suportat pel *daemon* `imapd` (els actuals suporten el protocol IMAP4rev1) permet accedir a un arxiu de correu electrònic (*mail file*) que es troba en una màquina remota. El servei `imapd` es presta mitjançant els ports 143 (imap2), 220 (imap3) o 993 (imaps) quan suporta encriptació per SSL. Si s'utilitza `inetd`, aquest servidor s'engega mitjançant una línia en el fitxer `/etc/inetd.conf` com:

```
imap2 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
imap3 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
```

En aquest exemple, es crida el *wrapper* `tcpd` que funciona amb `hosts.allow` i `hosts.deny` per a incrementar la seguretat. Les aplicacions més populars són `courier-imap`, `cyrus-imapd`, `dovecot-imapd`, entre d'altres. Per a provar que el servidor `imap` funciona, es podria utilitzar un client, per exemple Thunderbird/Icedove (Debian), Evolution, Squirremail, o qualsevol altre client que suporti IMAP, crear un compte per a un usuari local, configurar-lo

de manera adequada perquè es connecti sobre la màquina local i verificar el funcionament de `imap`.

Com a prova de concepte, podem fer la instal·lació de `apt-get install dovecot-imapd` que amb les opcions per defecte permet una connexió encriptada per SSL i sobre bústies *mailbox* (o si volem, sobre bústies *maildir* haurem de canviar la configuració de `/etc/dovecot/conf.d/10-mail.conf`). Dovecot és un servidor molt potent, per la qual cosa permet una gran quantitat d'opcions i configuracions (consulteu http://wiki2.dovecot.org/) i un resum aplicat d'aquestes en Debian Wheezy*. Les proves es poden completar configurant Evolution perquè es connecti al nostre servidor/usuari i llegir els correus del servidor prèviament configurat. És important notar que alguns clients de correu/servidors d'Imap només suporten el format *mailBox* i no *maildir*, i per aquest motiu s'ha de tenir en compte quan s'utilitzin els clients/servidors d'Imap. En les activitats que hem fet fins ara, tant `exim4` com `dovecot-imapd` suporten tots dos formats i s'han de configurar durant la instal·lació.

***https://workaround.org/ispmail/wheezy/setting-up-dovecot**

1.7.1. Aspectes complementaris

Suposem que com a usuaris, tenim quatre comptes de correu en servidors diferents i volem que tots els missatges que arriben a aquests comptes es recullin en un només, al qual puguem accedir externament, i que hi hagi també un filtre de correu brossa (*antispam*). Primer s'han d'instal·lar `exim4 + imapd` i comprovar que funcionen.

Per a recollir els missatges de diferents comptes, s'utilitzarà `Fetchmail`, (que s'instal·la amb `apt-get install fetchmail`). A continuació, s'ha de crear el fitxer `.fetchmailrc` en el nostre `$HOME` (també es pot utilitzar l'eina `fetchmailconf`), que haurà de ser una cosa semblant a:

```
set postmaster “adminp”
set bouncemail
set no spambounce
set flush
poll pop.domain.com proto pop3
   user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
poll mail.domain2.com
   user 'adminp' there with password 'MyPaSSwOrD' is 'adminp' here
   user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
```

L'acció `set` indica a `Fetchmail` que aquesta línia conté una opció global (enviament d'errors, eliminació dels missatges dels servidors, etc.). A continuació, s'especifiquen dos servidors de correu: un perquè comprovi si hi ha correu amb el protocol POP3 i un altre perquè provi a fer servir diversos protocols amb la finalitat de trobar-ne un que funcioni. Es comprova el correu de dos usuaris amb la segona opció de servidor, però tot el correu que es trobi s'envia a l'*spool* de correu de `adminp`. Això permet comprovar diverses bústies de diferents servidors, com si es tractés d'una única bústia. La informació específica de cada usuari comença amb l'acció `user`. El `Fetchmail` es pot posar en el

`cron` (per exemple, en `/var/spool/cron/crontabs/fetchmail` agregant `1 * * * * /usr/bin/fetchmail -s`) perquè s'executi automàticament o executar-lo en mode *daemon* (poseu `set daemon 60` en `.fetchmailrc` i executeu-lo una vegada, per exemple, en `autostart` de Gnome/KDE o en el `.bashrc` i s'executarà cada 60 segons).

Per a treure el correu brossa, es farà servir SpamAssassin.

Podeu instal·lar SpamAssassin mitjançant `apt-get install spamassassin`.

Aquesta configuració s'executarà mitjançant Procmail, que és una eina molt potent en la gestió del correu (permet repartir el correu, filtrar-lo, reenviar-lo de manera automàtica, etc.). Un cop instal·lat (`apt-get install procmail`), s'ha de crear un fitxer anomenat `.procmailrc` en el `home` de cada usuari, que cridarà l'SpamAssassin:

```
# Poseu yes per a missatges de funcionament o depuració
VERBOSE=no
# Considerem que els missatges estan en "~/.Maildir"), canviar si és un altre
PATH=/usr/bin:/bin:/usr/local/bin:.
MAILDIR=$HOME/Maildir
DEFAULT=$MAILDIR/

# Directori per a emmagatzemar els fitxers
PMDIR=$HOME/.procmail
# Comentar si no volem log de Procmail
LOGFILE=$PMDIR/log
# filtre de Smap
INCLUDERC=$PMDIR/spam.rc
```

L'arxiu `~/.procmail/spam.rc` conté:

```
# si l'SpamAssassin no és en el PATH, agregueu a la variable PATH el directori
:0fw: spamassassin.lock
| spamassassin -a

# Les tres línies següents mouran el correu spam a un directori anomenat
# "spam-folder". Si es vol guardar el correu en la safata d'entrada,
# per a després filtrar-lo amb el client, comenteu les tres línies.


:0:
* ^X-Spam-Status: Yes
spam-folder
```

L'arxiu `~/.spamassassin/user_prefs` conté algunes configuracions útils per a SpamAssassin (consulteu la bibliografia).

```
# user preferences file. Vegeu man Mail::SpamAssassin::Conf

# Llindar per a reconèixer un Spam.
# Default 5, però amb 4 funciona una mica millor
required_hits 4

# Llocs dels quals considerem que mai arribarà Spam
whitelist_from root@debian.org
whitelist_from *@uoc.edu

# Llocs dels quals sempre arriba SPAM (separats per comes)
blacklist_from viagra@dominio.com
```

```
# les direccions en Whitelist i Blacklist són patrons globals com:
# "amigo@lugar.com", "*@isp.net", o "*.domain.com".

# Inserteu la paraula SPAM en el subject (facilita fer filtres).
# Si no es desitja comentar la línia.
subject_tag [SPAM]
```

Això generarà un *tag* `X-Spam-Status: Yes` en la capçalera del missatge si es creu que el missatge és *spam*. Després, s'haurà de filtrar i posar a una altra carpeta o esborrar-lo directament. Es pot fer servir el `procmail` per a filtrar missatges de dominis, usuaris, etc. Finalment, es pot instal·lar un client de correu i configurar els filtres perquè seleccionin tots els correus amb `X-Spam-Status: Yes` i els esborri o els enviï a un directori. Després, verificarem els falsos positius (correus identificats com a brossa però que no ho són). Un aspecte complementari d'aquesta instal·lació és que si es desitja tenir un servidor de correu a través de correu web (*webmail*), és a dir, poder consultar els correus del servidor mitjançant un navegador sense haver d'instal·lar un client ni configurar-lo (com consultar un compte de Gmail o Hotmail), és possible instal·lar Squirrelmail (`apt-get install squirrelmail`) per a donar aquest servei.

Enllaç d'interès

Per a més informació sobre `procmail` i el filtrat de missatges, consulteu: http://www.debian-administration.org/articles/242.

Enllaç d'interès

Sobre Squirrelmail en Debian, consulteu: http://www.debian-administration.org/articles/200.

Enllaç d'interès

Hi ha altres possibilitats com instal·lar MailDrop en lloc de Procmail, Postfix en lloc d'Exim, o incloure Clamav/Amavisd com a antivirus (Amavisd permet vincular Postfix amb SpamAssassin i Clamav). Per a saber més sobre aquest tema, podeu visitar la següent pàgina web: http://www.debian-administration.org/articles/364.

1.8. Grups de discussió

Les *news* o grups de discussió són suportats mitjançant el protocol NNTP. Instal·lar un servidor de grups de discussió és necessari quan es desitja llegir *news* fora de línia, quan es vol tenir un repetidor dels servidors centrals o es vol un propi servidor mestre de *news*. Els servidors més comuns són INN o CNEWS, però són paquets complexos i destinats a grans servidors. Leafnode és un paquet USENET que implementa el servidor TNP, especialment indicat per a llocs amb grups reduïts d'usuaris, però on es desitja accedir a gran quantitat de grups de notícies. Aquest servidor s'instal·la en la configuració bàsica de Debian i es poden reconfigurar amb `dpkg-reconfigure leafnode` paràmetres com ara els servidors centrals, el tipus de connexió, etc. Aquest *daemon* s'engega des de `inetd` de manera similar al `imap` (o amb `xinetd`). Leafnode suporta filtres mitjançant expressions regulars indicades (del tipus `^Newsgroups:. *[,] alt.flame$`) en `/etc/news/leafnode/filters`, on per a cada missatge es compara la capçalera amb l'expressió regular i, si hi ha coincidència, el missatge es rebutja.

La configuració d'aquest servidor és simple i tots els arxius han de ser propietat d'un usuari de *news* amb permís d'escriptura (s'ha de verificar que aquest propietari existeix en `/etc/passwd`). Tots els arxius de control, *news* i la configuració es troben en `/var/spool/news`, excepte la configuració del propi servidor que està en el fitxer `/etc/news/leafnode/config`. En la configu-

ració, hi ha alguns paràmetres obligatoris que s'han de configurar (per exemple, perquè el servidor pugui connectar-se amb els servidors mestres), com són `server` (servidor de *news* des d'on s'obtindran i enviaran les *news*) i `expire` (nombre de dies als quals un fil o sessió s'esborrarà després d'haver estat llegit). Tenim, així mateix, un conjunt de paràmetres opcionals d'àmbit general o específic del servidor que podrien configurar-se. Per a més informació, consulteu la documentació (`man leafnode` o `/usr/doc/leafnode/README.Debian`). Per a verificar el funcionament del servidor, es pot fer `telnet localhost nntp` i, si tot funciona correctament, sortirà la identificació del servidor i es quedarà esperant una ordre. Com a prova, es pot introduir `help` (per a avortar, feu `Ctrl+[` i després `Quit`).

1.9. *World Wide Web* (httpd)

Apache és un dels servidors més populars i amb majors prestacions d'HTTP (*HyperText Transfer Protocol*). Apache té un disseny modular i suporta extensions dinàmiques de mòduls durant la seva execució. És molt configurable quant al nombre de servidors i de mòduls disponibles i suporta diversos mecanismes d'autenticació, control d'accés, *metafiles*, *proxy caching*, servidors virtuals, etc. Amb mòduls (inclosos en Debian) és possible tenir PHP3, Perl, Java Servlets, SSL i altres extensions*.

***Podeu consultar la documentació en http://www.apache.org.**

Apache està dissenyat per a executar-se com un procés *daemon standalone*. En aquesta forma, crea un conjunt de processos fills que gestionaran les peticions d'entrada. També pot executar-se com *Internet daemon* mitjançant `inetd`, per la qual cosa s'engegarà cada vegada que es rebi una petició, però no és recomanat. La configuració del servidor pot ser extremadament complexa segons les necessitats (consulteu la documentació); no obstant això, aquí veurem una configuració mínima acceptable. La seva instal·lació és simple, per exemple en Debian, `apt-get install apache2 apache2-doc apache2-utils`. La configuració del servidor estarà en */etc/apache2* i per defecte el *RootDirectory* en */var/www*. Després de la seva instal·lació, s'engegarà i posant com URL en un navegador http://localhost veurem que funciona (ens mostrarà el famós **It works!**). Hi ha 5 ordres que hauran d'estar a la ment de tot administrador:

- `a2enmod|a2dismod` per a habilitar/deshabilitar mòduls,
- `a2ensite|a2dissite` per a habilitar|deshabilitar llocs (virtuals) i
- `apachectl` per a gestionar la configuració del servidor (`start|stop|restart|graceful|graceful-stop|configtest|status|fullstatus|help`).

La configuració d'Apache2 en Debian és una mica diferent de la distribució general ja que intenta facilitar al màxim la configuració del servidor quant a mòduls, *hosts* virtuals i directives de configuració (no obstant això, ràpidament es poden trobar les equivalències amb altres distribucions). Els principals arxius que es troben en el directori */etc/apache2/* són *apache2.conf, ports.conf* i cinc directoris *mods-available|mods-enabled, sites-available|sites-enabled* i *conf.d*.

Per a informació addicional, podeu llegir */usr/share/doc/apache2.2** i en particular */usr/share/doc/apache2.2-common/README.Debian*.

1) *apache2.conf* és l'arxiu principal de configuració on es defineixen en un nivell funcional les prestacions del servidor i es criden els arxius de configuració corresponents (*ports, conf.d, sites-enabled*). Es recomana posar com a sufix *.load* per als mòduls que hagin de ser carregats i *.conf* per a les configuracions, però hi ha regles més extenses quant als sufixos/noms que poden ampliar-se en la documentació (p. ex., s'ignoren tots els arxius que no comencen per lletra o nom).

2) *ports.conf* (s'inclou en l'arxiu de configuració global) defineix els ports on s'atendran les connexions entrants, i quins d'aquests són utilitzats en els *hosts* virtuals.

3) Els arxius de configuració en *mods-enabled/* i *sites-enabled/* són per als llocs actius i els mòduls que desitgen ser carregats en el servidor. Aquestes configuracions s'activen creant un enllaç simbòlic des dels directoris respectius ***-available/* fent servir `a2enmod/a2dismod`, `a2ensite/a2dissite`.

4) Els arxius de *conf.d* són per a configuracions d'altres paquets o agregats per l'administrador i es recomana que acabin amb *.conf*.

5) Perquè sigui efectiva la configuració per defecte en aquests directoris, apache2 ha de ser gestionat a través de */etc/init.d/apache2* o `service` o `apache2ctl`.

6) L'arxiu *envvars* és el que contindrà la definició de les variables d'entorn i és necessari modificar bàsicament dos USER/GROUP que seran amb les quals s'executarà i obtindrà els permisos. Per defecte es crea l'usuari www-data i el grup www-data (es poden canviar). Per aquest motiu, s'haurà de fer servir `APACHE_RUN_USER=www-data` i `APACHE_RUN_GROUP=www-data`.

Apache també pot necessitar integrar diversos mòduls en funció de la tecnologia que suporti, per la qual cosa s'hauran d'agregar les biblioteques/paquets corresponents, per exemple:

1) Perl: `apt-get install libapache2-mod-perl2`

2) Rugby: `apt-get install libapache2-mod-ruby`

3) `Python: apt-get install libapache2-mod-python`

4) MySQL in Python: `apt-get install python-mysqldb`

5) PHP: `apt-get install libapache2-mod-php5 php5 php-pear php5-xcache`

6) PHP with MySQL: `apt-get install php5-mysql`

1.9.1. Servidors virtuals

Per *servidors virtuals* s'entenen els llocs aliats que seran servits cadascun de manera independent de l'altre amb els seus propis arxius i configuració. En primer lloc, deshabilitarem el lloc per defecte amb `a2dissite default`. Els llocs que crearem seran remix.world i lucix.world, que disposaran de dos arxius de configuració en */etc/apache2/sites-available/* anomenats com el domini.

Contingut de l'arxiu `/etc/apache2/sites-available/remix.world.conf`

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName remix.world
    ServerAlias www.remix.world
    DocumentRoot /var/www/remix/
    ErrorLog /var/log/apache2/remix-error.log
    CustomLog /var/log/apache2/remix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Contingut de l'arxiu `/etc/apache2/sites-available/lucix.world.conf`

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName lucix.world
    ServerAlias www.lucix.world
    DocumentRoot /var/www/lucix/
    ErrorLog /var/log/apache2/lucix-error.log
    CustomLog /var/log/apache2/lucix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Aquesta configuració és molt bàsica i l'estudiant haurà de consultar la informació detallada en [14]. Com es pot observar, els directoris arrel per a cada domini estaran en */var/www/remix|lucix* i els arxius de *log* en /errors/accessos en */var/log/apache2/mmmm-error.log* i *var/log/apache2/nnnn-access.log/*. Per a crear els directoris `mkdir -p /var/www/remix`; `mkdir -p /var/www/lucix` i en els quals es podria posar un *index.html* amb alguna identificació que mostrés quin domini s'està carregant. Per exemple, per a remix.world:

```
<html><body><h1>REMIX: It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
</body></html>
```

I el mateix per a lucix.world, però canviant la línia en <h1></h1>. Per als *logs* no hem de fer res ja que el directori */var/log/apache2* ja existeix i els arxius els crearà el servidor. Finalment, hem d'activar els llocs (crear l'enllaç des de *sites-available* a *sites-enabled*) amb `a2ensite remix.world.conf`; `a2ensite lucix.world.conf` i reiniciar apache2 amb `service apache2 reload`. Com que no disposem dels dominis en un DNS primari, podem editar */etc/hosts* i agregar per a la IP del nostre servidor (p. ex., 192.168.1.37) dues línies:

```
192.168.1.37 remix.world
192.168.1.37 lucix.world
```

Després, des d'un navegador podrem introduir l'URL remix.world i el resultat serà la visualització de l'index.html que ens dirà: **REMIX: It works!**

Un dels avantatges d'Apache és que pot agregar funcionalitat mitjançant mòduls especialitzats i que es trobaran en */etc/apache2/mods-available/*. Per a obtenir la llista de mòduls disponibles per a Apache podem fer, per exemple, `apt-cache search libapache2*`, i per a instal·lar-lo `apt-get install [module-name]`, els quals estaran disponibles per al seu ús (recordeu que pot ser necessària alguna configuració addicional en els arxius del lloc). Podem mirar els disponibles amb `ls -al /etc/apache2/mods-available/` i instal·lar-lo amb `a2enmod [module-name]`. Per a posar en una llista els mòduls carregats, podem fer servir `/usr/sbin/apache2 -M` que ens posarà en una llista amb *shared* els carregats dinàmicament i amb *static* els que es troben compilats amb el servidor (aquests es poden obtenir també amb `apache2 -l`). Els mòduls en el directori *mods-available* tenen extensions *.load* (indica la biblioteca que cal carregar) i *.conf* (configuració addicional del mòdul), però quan utilitzem l'ordre `a2enmod` només s'ha d'indicar el nom del mòdul sense extensió. Per a deshabilitar un mòdul, `a2dismod [module-name]`.

Com a mostra d'aquestes propietats, configurarem un lloc segur (https) sota el domini remix.world però que redirigirem al directori */var/www/remix.ssl*. En primer lloc, crearem un certificat (autosignat) per al nostre lloc amb

```
make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/private/remix.crt
```

indicant-hi el domini que volem validar (remix.world en el nostre cas) –només cal introduir el domini i deixar els àlies en blanc– i si no podem executar `make-ssl-crt`, hem d'assegurar-nos que tenim el paquet `ssl-cert`. Després activem el mòdul SSL amb `a2enmod ssl`, creem el directori */var/www/remix.ssl* i modifiquem l'index.html com vam fer amb els anteriors. Després, creem la configuració del lloc (podem utilitzar la que ve per defecte i modificar-la):

```
cd /etc/apache2/sites-available; cp default-ssl remix.world.ssl.conf
```

Editem l'arxiu *remix.world.ssl.conf* (només mostrem les línies principals/modificades):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin adminpSySDW.nteum.org
    DocumentRoot /var/www/remix.ssl
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/remix.ssl>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
# línies igual que l'arxiu original...
```

```
    ErrorLog $APACHE_LOG_DIR/remix.world.ssl_error.log
    CustomLog $APACHE_LOG_DIR/remix.world.ssl_access.log combined

    SSLEngine on
    SSLCertificateFile /etc/ssl/private/remix.crt
    #SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
# línies igual que l'arxiu original...
</VirtualHost>
</IfModule>
```

Finalment, ens queda activar el lloc (`a2ensite remix.world.ssl.conf`), reiniciar Apache2 (amb `service apache2 reload`) i des del navegador fer *https://remix.world* que, com que el certificat és autosignat, ens farà un advertiment i acceptarem el certificat i haurem d'obtenir **SSL - REMIX: It works!**

Un aspecte interessant és la funció de l'arxiu *.htaccess** en els directoris del nostre domini. Aquest arxiu es pot utilitzar per al control d'accés al lloc (p. ex., habilitar/restringir IP), control d'accés a carpetes, llistes, reencaminaments (p. ex., a una altra pàgina/*site*, a una altra carpeta, a un altre domini, a https, etc.), evitar el *hotlinking* (per a evitar que ens facin enllaços a fitxers, generalment vídeos, que consumeixen amplada de banda del nostre servidor), canviar la pàgina per defecte, crear URL amigables, afavorir el cau del nostre lloc, etc. Com a mostra d'això, per a evitar per exemple que una carpeta sigui inaccessible, n'hi ha prou amb posar un arxiu *.htaccess* en la mateixa amb el contingut `deny from all`. Per a permetre que una carpeta del nostre lloc (p. ex., del domini remix.com/valid-user) tingui accés amb un usuari i passwd, haurem de crear dins d'aquesta un arxiu *.htaccess* amb el següent contingut (també podem crear un index.html modificat dins d'aquesta carpeta per a verificar-ne el funcionament):

***http://httpd.apache.org/docs/2.2/howto/htaccess.htm**

```
AuthName "Restricted Area"
AuthType Basic
AuthUserFile /etc/apache2/htpasswd
AuthGroupFile /dev/null
require valid-user
```

Per a crear l'usuari, fem `htpasswd -c /etc/apache2/htpasswd adminp` que ens demanarà el *passwd* per a aquest usuari i l'emmagatzemarà en l'arxiu indicat. Després, posem com a URL *http://remix.world/valid-user/*. Ens demanarà l'usuari (adminp) i el *passwd* que emmagatzemem i veurem **REMIX->Valid-User: It works!** En cas contrari, ens continuarà demanant l'usuari /*passwd* i si fem *Cancel* no indicarà un missatge d'*Authorization Required* impedint l'accés.

1.9.2. Apache + PHP + Mysql + PhpMyAdmin

Una qüestió important per als servidors web dinàmics és aprofitar els avantatges d'Apache PHP (un llenguatge de programació usat generalment per a la creació de contingut per a llocs web) i una base de dades com MySQL incloent

un programa administrador de MySQL com PHPMyAdmin, tot això funcionant conjuntament. Les distribucions han evolucionat molt i en Debian és summament fàcil engegar aquest conjunt (però tampoc representa cap dificultat descarregar-se el programari font, compilar-lo i instal·lar-lo si es desitja, per exemple, tenir les últimes versions dels paquets per algun motiu però recordeu que implicarà més treball i dedicació).

En primer lloc, instal·lem PHP amb `apt-get install php5`, que ens indicarà que canviarà la manera de treballar d'Apache2 instal·lant-ne un altre. Això es produeix perquè la configuració per defecte d'Apache treballa amb una eina anomenada *MPM-worker*. Aquest és un mòdul de multiprocessament que pot manejar múltiples peticions ràpidament utilitzant múltiples *threads* per procés client. No obstant això, aquest no és compatible amb algunes extensions PHP i per això l'*MPM-worker* és reemplaçat per *MPM-prefork*, que permet manejar totes les peticions PHP (en mode compatibilitat) i evitar que si una petició falla pugui afectar altres peticions. Hi ha un altre mòdul anomenat `mpm-itk` (http://mpm-itk.sesse.net/) que és similar a *prefork* però té millors prestacions i gestió de permisos (consulteu la bibliografia en apache.org). Per a verificar que PHP funciona, creem un fitxer per exemple dins de *RootDirectory* de remix.world anomenat *test.php* amb el següent contingut: *<?php phpinfo() ?>*, i si en l'URL introduïm *http://remix.world/test.php* haurem de veure una taula amb la versió i informació sobre el paquet PHP instal·lat.

Per instal·lar els paquets MySQL i PHPMyAdmin, farem `apt-get install mysql-server` (és molt important que recordeu la contrasenya d'accés que introduïm, però sempre podem fer `dpkg-reconfigure mysql-server`; tenint en compte que perdrem tot el que hi hagi en la BD). També hi ha altres mètodes (menys agressius) per a recuperar la contrasenya del *root*). Després, per a instal·lar PHPMyAdmin farem `apt-get install phpmyadmin` i prestar atenció, ja que ens demanarà la clau d'accés per a entrar en la base de dades i crear una clau d'accés per a entrar en l'aplicació via navegador. Després, podrem posar en l'URL del nostre navegador *http://localhost/phpmyadmin*, ens sol·licitarà l'usuari (*root* generalment) i el *passwd* que hem introduït i ja podrem gestionar el servidor de bases de dades MySQL.

1.9.3. Altres servidors httpd

Lighttpd és un servidor web (amb llicència BSD) dissenyat per a ser ràpid, segur, flexible, que implementa la majoria d'estàndards i està optimitzat per a entorns on la velocitat és molt important (consumeix menys CPU/RAM que altres servidors) i és molt apropiat per a qualsevol servidor que hagi de donar suport a grans càrregues. Entre les seves principals característiques, hi ha la de *virtual hosting*, reencaminaments http i reescriptures d'URL, donar suport a CGI, SCGI i FastCGI, PHP, Ruby, Python entre d'altres i a més amb consum de memòria constant.

La seva instal·lació en Debian és `apt-get install lighttpd`, i si tenim Apache sobre el port 80 ens donarà un error. Per a això, haurem d'editar l'arxiu */etc/lighttpd/lighttpd.conf* i canviar la línia `server.port = 8080` i reiniciar `service lighttpd start`. Des del navegador, es pot escriure l'adreça *http://localhost:8080index.lighttpd.html* i llavors veurem la pàgina inicial de lighttpd. Per defecte, Lighttpd té el seu directori arrel en */var/www* (en Debian) i l'arxiu de configuració en */etc/lighttpd/lighttpd.conf*. Configuracions addicionals són en */etc/lighttpd/conf-available* i poden ser habilitades amb l'ordre `lighttpd-enable-mod`, la qual crea enllaços entre *conf-enabled* i *conf-available*. Es poden deshabilitar amb `lighttpd-disable-mod`.

Per a habilitar el servidor de FastCGI per a executar PHP, haurem d'instal·lar PHP-FPM amb la instrucció `apt-get install php5-fpm php5` i sobre l'arxiu */etc/php5/fpm/php.ini* treure el comentari a la línia `cgi.fix_pathinfo=1`. Després haurem d'activar el servidor PHP-FPM, per la qual cosa farem una còpia de l'arxiu original i el modificarem:

```
cd /etc/lighttpd/conf-available/
cp 15-fastcgi-php.conf 15-fastcgi-php-spawnfcgi.conf
```

Modificar *15-fastcgi-php.conf* amb:

```
# -*- depends: fastcgi -*-

# Start an FastCGI server for php
fastcgi.server += ( ".php" =>
        ((
                "socket" => "/var/run/php5-fpm.sock",
                "broken-scriptfilename" => "enable"
        ))
)
```

Per a habilitar fastcgi, haurem de carregar els mòduls `lighttpd-enable-mod fastcgi` i `lighttpd-enable-mod fastcgi-php`, la qual cosa ens crea els enllaços corresponents, que amb la instrucció `ls` podem visualitzar: `ls -l /etc/lighttpd/conf-enabled`. Després, podem reiniciar amb la instrucció `service lighttpd force-reload`. Per a visualitzar si el servidor i FastCGI funcionen, creem un arxiu */var/www/info.php* amb el següent contingut `<?php phpinfo(); ?>` i podrem visualitzar la pàgina de configuració de PHP on indica com Server API = FPM/FastCGI (http://localhost:8080/info.php).

Un altre servidor molt utilitzat actualment és **Nginx** (http://nginx.org/) programat en C i llicència BSD. Les seves funcions principals són com a servidor web/*proxy* invers de molt alt rendiment (pot suportar més de 10.000 connexions simultànies) i també pot funcionar com a *proxy* per a protocols de correu electrònic (IMAP/POP3). És un servidor utilitzat per grans instal·lacions

(WordPress, Netflix, Hulu, GitHub i parts de Facebook, entre d'altres) i entre les seves principals característiques hi ha (a més de servidor d'arxius estàtics, índexs i autoindexat i *proxy* invers amb opcions de cau) el balanç de càrrega, tolerància a fallades, SSL, FastCGI, servidors virtuals, *streaming* d'arxius (FLV i MP4.8) i suport per a autenticació, compatible amb IPv6 i SPDY. La seva instal·lació bàsica és simple, i per a la seva configuració, es pot consultar la wiki de `nginx` en http://wiki.nginx.org/configuration.

1.10. Servidor de WebDAV

El nom WebDAV són les sigles de *Web Based Distributed Authoring and Versioning* (també es refereix al grup de treball d'*Internet Engineering Task Force*) i és un protocol que permet que el web es transformi en un mitjà llegible i editable i proporciona funcionalitats per a crear, canviar i moure documents en un servidor remot (típicament, un servidor web). Això s'utilitza sobretot per a permetre l'edició dels documents que envia un servidor web, però també es pot aplicar a sistemes d'emmagatzematge generals basats en el web i als quals es pot accedir des de qualsevol lloc. En aquest subapartat, instal·larem un servidor WebDAV sobre Apache. El procés és el següent:

Enllaç d'interès

Sobre la integració de WebDAV amb Apache, podeu consultar l'article "WebDAV on Apache2" disponible en: http://www.debian-administration.org/articles/285

1) Verificar que tenim instal·lat Apache2 i, si no, fer la seva instal·lació com hem vist anteriorment i verificar que funciona (`apt-get install apache2`).

2) Habilitar els mòduls d'Apache que són necessaris per a WebDAV: `a2enmod dav_fs` i `a2enmod dav`.

3) Crear el directori per al directori virtual (podem fer, per exemple, `mkdir -p /var/www/webdav`) i permetre que Apache sigui el propietari del directori `chown www-data /var/www/webdav/`

4) Crear l'arxiu */etc/apache2/sites-available/webdav.conf* per a definir la funcionalitat del servidor (en aquesta configuració, estem configurant tot el servidor com a WebDAV però podria ser un servidor virtual i en mode SSL per a major seguretat):

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    DocumentRoot /var/www/webdav/
    ErrorLog /var/log/apache2/webdav-error.log
    CustomLog /var/log/apache2/webdav-access.log combined
   <Directory /var/www/webdav>
       Options Indexes MultiViews
       AllowOverride None
       Order allow,deny
       allow from all
       DAV On
       AuthName "Restricted WeDav Area"
       AuthType Basic
       AuthUserFile /etc/apache2/htpasswd
       AuthGroupFile /dev/null
       require valid-user
   </Directory>
</VirtualHost>
```

Es pot comprovar que la configuració és correcta amb

`apache2ctl configtest`.

5) Com s'ha fet anteriorment, es creen els usuaris indicant el -c si és el primer usuari que cal crear (`htpasswd [-c] /etc/apache2/htpasswd` usuari).

6) Es reinicia Apache perquè llegeixi la configuració `/etc/init.d/apache2 reload` i ja ens podem connectar a `http://localhost`, després de fer l'autenticació.

7) Des d'un GNU/Linux, podem provar la funcionalitat del servidor obrint el `nautilus` (gestor de fitxers) i des del menú *File->Connect to Server* podem seleccionar Servidor WebDAV introduint les dades (IP, directori, usuari, passwd) i ja tindrem accés com si d'una carpeta local es tractés.

8) Des de MacOS, podem utilitzar el mateix procediment que l'anterior des del gestor d'arxius o instal·lar un client específic (igualment per a Windows). El més recomanat per a això és CyberDuck (http://cyberduck.io/), que té llicència GPL i és una excel·lent aplicació (suporta múltiples protocols) i molt fàcil de configurar.

9) Una altra forma de provar-ho és amb un client WebDAV (en mode text), per exemple Cadaver*, amb `apt-get install cadaver`. A continuació, ens connectem al servidor amb `cadaver IP-nom del servidor`, i després d'autenticar-nos, podem crear un directori (`mkdir`), editar un arxiu, fer la llista d'un directori (`ls`), canviar de directori (`cd`), canviar els permisos d'execució (`chexec`), esborrar-lo (`rm`), etc.

*http://www.webdav.org/cadaver

En moltes ocasions, i atès que estarem fent transferències d'arxius, és important preservar la privadesa, per la qual cosa seria adequat treballar amb WebDAV però sobre SSL. La seva configuració no implica majors complicacions i veurem una manera diferent de generar els certificats (seran autosignats, fins que puguem tenir la nostra pròpia entitat certificadora) per a un domini en particular, en el nostre cas webdav.world. Per a això fem:

1) Ens canviem al directori on emmagatzemarem els certificats:

```
cd /etc/ssl/private
```

Fem la petició del certificat:

```
openssl req -config /etc/ssl/openssl.cnf -new -out webdav.csr
```

Aquesta ordre ens demanarà un *passwd* i una sèrie d'informació que quedarà en el certificat, però la més important és *Common Name* (CN), que serà on validarà el certificat (en el nostre cas, webdav.world).
Podem verificar la petició amb:

```
openssl req -in /etc/ssl/private/webdav.csr -noout -text
```

2) Creem la clau (*key*):

```
openssl rsa -in privkey.pem -out webdav.key
```

3) Signem:

```
openssl x509 -in webdav.csr -out webdav.crt -req -signkey webdav.key
-days 3650
```

Ho podem verificar amb:

```
openssl x509 -noout -in /etc/ssl/private/webdav.crt -text
```

4) Generem un certificat en format DER: `openssl x509 -in webdav.crt -out webdav.der.crt -outform DER` Es pot verificar amb: `openssl x509 -in /etc/ssl/private/webdav.der.crt -inform der -noout -text`

5) Ara, generem l'arxiu de configuració d'Apache a partir de l'anterior: `cd /etc/apache2/sites-available; cp webdav.conf webdav-ssl.conf` Modifiquem per a incloure les següents quatre línies a l'inici i modificar el VirtualHost:

```
<VirtualHost *:443>
ServerName webdav.world
SSLEngine on
SSLCertificateFile /etc/ssl/private/webdav.crt
SSLCertificateKeyFile /etc/ssl/private/webdav.key
...
```

Només ens queda activar el lloc (`a2ensite webdav-ssl.conf`), reiniciar Apache (`service apache2 restart`) i verificar que funciona en l'adreça https://webdav.world/ prèvia acceptació del certificat (recordar posar una entrada en */etc/hosts* amb el nom del domini i l'IP de la màquina similar a com es va fer en remix.world).

1.11. Servei de *proxy*: Squid

Un servidor *proxy* (*proxy server*, PS) s'utilitza per a estalviar amplada de banda de la connexió de xarxa, millorar la seguretat i incrementar la velocitat per a obtenir pàgines de la xarxa (*web-surfing*).

Squid és un *proxy caching server* per a web i dóna suport als protocols HTTP, HTTPS, FTP, entre d'altres. Aquest redueix l'amplada de banda, millora el temps de resposta emmagatzemat en cau i reutilitza les pàgines més freqüents. Squid té un extens conjunt de regles de control que permeten optimitzar el flux de dades entre el client i el servidor aportant seguretat i control i encaminant les peticions correctament, la qual cosa permet el seu control i millora la utilització de l'amplada de banda de la xarxa. Squid té diferents modes de funcionament, però com a més importants podem esmentar *Forward Proxy* (és la manera bàsica sobre la qual es configura tota la resta), *Transparent* o *Interception Proxy* (permet incloure un *proxy* en una xarxa sense que els clients hagin de configurar res) i *Reverse Proxy* o *Accelerator-mode* (permet executar Squid per a millorar la resposta d'una granja de servidors web).

Per a instal·lar Squid com a *proxy-cache* (http://www.squid-cache.org/), en Debian fem `apt-get install squid3` i editarem l'arxiu de configuració */etc/squid3/squid.conf* per a portar a terme una configuració bàsica. S'ha de tenir en compte que Squid és molt potent, però això es tradueix en una configuració que pot ser complexa; com a idea, simplement hem de considerar que l'arxiu de configuració, que està molt ben explicat, té aproximadament 5.700 línies (no obstant això, si executem

```
grep -v "^#" /etc/squid3/squid.conf | awk '$1 != "" {print $0}'
```

podrem veure que la configuració bàsica són unes 40 línies mentre que la resta són comentaris i opcions comentades).[16][17]

Definir l'ACL (*access control list*) per a habilitar la xarxa/IP que desitgem fer de *proxy*, en la línia 718 agreguem: `acl lan src 192.168.1.0/24`

Permetre l'accés, en la línia 842 (aprox.) agreguem: `http_access allow lan`
Canviar el port d'accés (línia 1136), per defecte és `http_port 3128` i es pot deixar que sigui l'estàndard per a Squid o posar el que es prefereixi (per exemple, 8080).

Agregar les regles de control, en la línia 3449 (aprox.):

```
request_header_access Referer deny all
request_header_access X-Forwarded-For deny all
request_header_access Via deny all
request_header_access Cache-Control deny all
```

Definim el nom visible, línia 3748 (aprox.): `visible_hostname remix.world`

Modifiquem la visibilitat de l'IP, línia 5541 (aprox.): `forwarded_for off`

Finalment, reiniciem el servidor: `service squid3 restart`
Ens donarà un missatge similar a *[ok] Restarting Squid HTTP Proxy 3.x: squid3[....] Waiting.....done.* (trigarà uns segons).

Amb el servidor en marxa, podem configurar els navegadors perquè utilitzin el *proxy*. Per exemple, en IceWeasel/Firefox en l'apartat de *Preferences/Options->Advanced->Network->Proxy* podem introduir les dades del nostre servidor. Sobre Chrome, per exemple en Windows, s'ha d'anar a *Settings->Advanced->Change proxy setting->Connections->Lan Settings* i introduir les dades del nostre servidor.

Una altra de les opcions que permet Squid és actuar com a *reverse proxy*, que és un tipus de *proxy* on les dades es recuperen d'un servidor i retornen als clients com si s'originessin en el *proxy*, de manera que els servidors queden ocults als clients com es mostra en la figura*. Això permet fer polítiques de balanç de

*http://upload.wikimedia.org/wikipedia/commons/6/67/reverse_proxy_h2g2bob.svg

càrrega i que les peticions estiguin en un únic domini, malgrat que internament poden estar distribuïdes en diversos servidors. Per a la seva configuració, hem de fer:

Especificar l'adreça del servidor intern, en la línia 1136 modificar:

```
http_port 80 defaultsite=192.168.1.33
```

Agregar cache_peer, en la línia 1937 (aprox.) agregar:

```
cache_peer 192.168.1.33 parent 80 0 no-query originserver
```

Canviar l'acl per a permetre qualsevol connexió, en la línea 842 (aprox.) modificar `http_access allow all`

Finalment, reiniciem el servidor: `service squid3 restart`

Quan ens connectem a l'IP/domini del servidor *proxy*, en realitat veurem les pàgines enviades pel servidor 192.168.1.33. Com a prova de concepte (si volem fer la prova amb una única màquina), podem posar, en lloc del servidor 192.168.1.33, un servidor extern (p. ex., debian.org o la seva IP), i quan posem com a URL el del nostre domini, visualitzarem la pàgina de debian.org.

Altres de les configuracions àmpliament utilitzades és *interception proxy* (o *transparent proxy*), que intercepta la comunicació normal a la capa de xarxa sense necessitat de configuracions específiques en el client, i per la qual cosa no sabran que estan darrere d'un *proxy*. Generalment, un *transparent proxy* està normalment localitzat entre el client i Internet amb el *proxy* fent les funcions de *router* o *gateway*. És habitual en les institucions que desitgen filtrar algun trànsit dels seus usuaris, ISP per a fer cau i estalviar amplada de banda o països que controlen l'accés a determinats llocs per part dels seus ciutadans. Per a implementar-lo sobre una institució, l'habitual és disposar d'una màquina amb dues interfícies de xarxa amb una de connectada a la xarxa interna i una altra cap a la xarxa externa. Els passos per a la seva configuració estan descrits en http://wiki.squid-cache.org/configexamples/intercept/linuxdnat:

Sobre *squid.conf*:

Modificar la `acl/http_access` sobre les IP, IP/Mask permeses com en el primer cas.

Configurar el port/mode com `http_port 3129 transparent` o en Squid 3.1+ s'haurà d'utilitzar `http_port 3129 intercept` per a interceptar paquets DNAT.

Sobre */etc/sysctl.conf* modificar:

Permetre el *packet forwarding*: `net.ipv4.ip_forward = 1`

Controlar el *source route verification*: `net.ipv4.conf.default.rp_filter = 0`

No acceptar *source routing*: `net.ipv4.conf.default.accept_source_route = 0`

Considerant que l'IP de *proxy* està en la variable SQUIDIP i el port està en SQUIDPORT, incloure les següents regles de DNAT:

```
iptables -t nat -A PREROUTING -s $SQUIDIP -p tcp -dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp -dport 80 -j DNAT -to-destination
     $SQUIDIP:$SQUIDPORT
iptables -t nat -A POSTROUTING -j MASQUERADE
iptables -t mangle -A PREROUTING -p tcp -dport $SQUIDPORT -j DROP
```

1.11.1. *Proxy* SOCKS

SOCKS (abreujament de SOCKetS) és un protocol d'Internet (en el model OSI estaria en una capa intermèdia entre la d'aplicació i la de transport) que permet a les aplicacions en mode client-servidor travessar de manera transparent un *firewall* de xarxa. Els clients que hi ha darrere d'un *firewall*, els quals necessiten accedir als servidors de l'exterior, poden connectar-se en el seu lloc a un servidor *proxy* SOCKS. Aquest servidor *proxy* controla quin client pot accedir al servidor extern i passa la petició al servidor. SOCKS pot ser utilitzat també de la forma contrària, la qual cosa permet que els clients de fora del *firewall* (clients externs) es connectin als servidors de dins del *firewall* (servidors interns).[26][27]

S'ha de tenir en compte que SOCKS només serveix en mode client/servidor, per la qual cosa un usuari ha de tenir instal·lat un client SOCKS, ja sigui en l'aplicació (com Firefox, Chrome) o dins de la pila TCP/IP des d'on el programari del client redirigeix els paquets en un túnel SOCKS. El procediment habitual comença quan el client SOCKS (p. ex., intern en una xarxa privada) inicia una connexió a un servidor SOCKS (el protocol SOCKS permet l'autenticació i el registre de les sol·licituds de connexió) i aquest (servidor SOCKS) actuar com a client IP per a la sol·licitud de connexió (del client intern en nom seu), la qual cosa significa que el servidor extern només serà conscient de les peticions del servidor SOCKS (així que actuarà com a *proxy forwarding*).

La pregunta habitual és si SOCKS resol de manera diferent el problema d'accés extern mitjançant NAT. La resposta és sí, ho fa de manera diferent, ja que els paquets en NAT només modifiquen les adreces (p. ex., canvien les IP privades per les pública del *router* com succeeix en un *router* ADSL) i el servidor rep/contesta les peticions. La sessió IP s'estableix directament des del client al servidor i el *router*/FW només modifica/filtra el paquet però no hi ha autenticació ni inspecció del paquet/aplicació/dades (es podria fer, però és complex). Els avantatges de SOCKS rauen en el fet que proveeix autenticació per a protocols que no ho permeten, pot traspassar el *routing* per defecte d'una xarxa interna. Si bé HTTP i Telnet suporten autenticació per *firewall* (p. ex., utilitzant *Authenticated Proxy on* a Cisco *firewall*), els protocols encriptats mai poden ser autenticats per un FW, i en canvi SOCKS si pot fer-ho. No obstant això, hi ha

desavantatges, ja que el client ha de tenir interfície a SOCKS, el SO client ha de tenir interfície a SOCKS (per a interceptar el trànsit i reexpedir-lo al SOCKS *proxy*) i necessitarem un servidor SOCKS específic per a aquesta fi.

Un cas d'ús habitual és si considerem que estem en un punt de connexió sense fil oberta (p. ex., Wi-Fi) i no es desitja enviar dades de navegació sobre text net o es vol accedir a una pàgina web filtrada per *router*/FW perimetral. Una solució molt simple és utilitzar SSH (que inclou un servidor SOCKS) que pot xifrar tot el trànsit de navegació pel web i reencaminar-lo a través d'un equip de confiança quan s'està en algun altre punt de la xarxa. Per a això, haurem de disposar d'un servidor SSH perquè actuï com a representant en un ordinador remot (que li permeti connectar-se al mateix a través de SSH) i un client d'SSH en l'equip que està utilitzant. El que es farà amb un *proxy* és la creació d'un *middle-person* entre l'usuari i Internet. El navegador farà les sol·licituds de pàgines web al servidor *proxy*, que controla la sol·licitud i obté la pàgina des d'Internet i les retorna al client. El lloc web en realitat pensa que la sol·licitud prové del servidor *proxy*, no de l'equip que l'ha originat ocultant l'adreça IP d'origen. A més, la connexió entre l'ordinador i el *proxy* que passa a través d'SSH és xifrada i això evita que algú pugui obtenir els paquets des de la Wi-Fi (*sniffers* de Wi-Fi) en el lloc de la connexió.

Per a la seva configuració des d'on ens connectem, hem de tenir accés a un servidor SSH, sobre el qual crearem un túnel que passarà el trànsit web entre la nostra màquina local i el *proxy* SOCKS sobre SSH. Per a això, executem sobre la nostra màquina `ssh -ND 9999 login@remote-server.org` on haurem de reemplaçar login@remote-server.org amb l'usuari i nom o IP del servidor remot. El que està fent aquesta ordre és un *port forwarding* a través del port 9999 (pot ser qualsevol altre, però convé que sigui superior a 1024 per a evitar que només ho pugui fer *root*) i la connexió es reenvia a través d'un canal segur on el protocol d'aplicació s'utilitza per a determinar on connectar des de la màquina remota. Actualment, OpenSSH suporta els protocols SOCKS4-5 i per això actuarà com un servidor SOCKS. A continuació, se sol·licitarà el *passwd* i una vegada autenticat no passarà res (el -N indica que no obri un *prompt* interactiu, però continuarà funcionant). Si pel *firewall* només podem sortir pel port 443, per exemple, hauríem de configurar el *ssh server* per a escoltar pel port 443 i en el seu lloc executar `ssh -ND 9999 login@remote-server.org -p 443`. Ara és necessari configurar el client per a connectar-se al *proxy*, per exemple Firefox: *Options -> Advanced -> Network -> Connection* i seleccionar SOCKS, com a nom del servidor *localhost* (o el seu nom real si en té) i el port (9999) i guardar els ajustos i verificar que podem navegar sense problemes. Es pot utilitzar el *plugin* FoxyProxy* per a Firefox, que permet canviar entre el *proxy* i la connexió directa en funció del lloc o d'un control. Com a mesura addicional (d'anonimat), es pot configurar el servidor *proxy* per a resoldre peticions DNS en lloc del mètode habitual en Firefox posant com a URL `about:config` i modificant `network.proxy.socks_remote_dns=true`. També per a connexions lentes es pot utilitzar l'opció -C de `ssh` per a fer servir la compressió de SSH per `gzip`. En Thunderbird o altres clients, la configuració és similar.

***https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard**

Si el túnel deixa de funcionar (acostuma a ocórrer en xarxes molt ocupades), es pot utilitzar el paquet `autossh` en lloc del `ssh` per a establir la connexió que s'encarregarà de mantenir el túnel obert reiniciant automàticament la connexió. Un altre paquet interessant és `tsocks` (http://tsocks.sourceforge.net/), que es pot utilitzar quan el client que desitgem utilitzar no suporta el protocol SOCKS. `tsocks` monitora la trucada d'inici de sessió d'una aplicació (*connect*) i redirecciona la comunicació cap al *server* SOCKS sense que l'aplicació tingui cap informació. Per a això, s'ha d'instal·lar `tsocks` i configurar el *proxy* SOCKS que haurà d'utilitzar en el fitxer */etc/tsocks.conf* indicant-hi els valors (p. ex., server = 127.0.0.1, server_type = 5, server_port = 9999). Després, bastarà a cridar l'aplicació amb `tsocks aplicació` o simplement l'ordre que obrirà una nova *shell* redirigida al *proxy* i per la qual cosa tot el que s'executi allà serà enviat al *proxy* SOCKS.

1.12. OpenLdap (LDAP)

LDAP significa *Lightweight Directory Access Protocol* i és un protocol per a accedir a dades basades en un servei X.500. Aquest s'executa sobre TCP/IP i el directori és similar a una base de dades que conté informació basada en atributs. El sistema permet organitzar aquesta informació de manera segura i utilitza rèpliques per a mantenir la seva disponibilitat, la qual cosa assegura la coherència i la verificació de les dades accedides-modificades.

El servei es basa en el model client-servidor, on existeixen un o més servidors que contenen les dades; quan un client es connecta i sol·licita informació, el servidor respon amb les dades o amb un punter a un altre servidor d'on podrà extreure més informació; no obstant això, el client només veurà un directori d'informació global [28, 19]. Per a importar i exportar informació entre servidors `ldap` o per a descriure una sèrie de canvis que s'aplicaran al directori, el format utilitzat es diu LDIF (LDAP *Data Interchange Format*). LDIF emmagatzema la informació en jerarquies orientades a objectes que després seran transformades al format intern de la base de dades. Un arxiu LDIF té un format similar a:

```
dn: o = UOC, c = SP o: UOC
objectclass: organization
dn: cn = Remix Nteum, o = UOC, c = SP
cn: Remix Nteum
sn: Nteum
mail: nteumuoc.edu
objectclass: person
```

Cada entrada s'identifica amb un nom indicat com a `DN` (*Distinguished Name*). El DN consisteix en el nom de l'entrada més una sèrie de noms que el relacionen amb la jerarquia del directori i on existeix una classe d'objectes (`objectclass`) que defineix els atributs que es poden fer servir en aquesta entrada. LDAP proveeix d'un conjunt bàsic de classes d'objectes: **grups** (inclou llistes desordenades d'objectes individuals o grups d'objectes), **localitzacions**

(com països i la seva descripció), **organitzacions i persones**. Una entrada pot, a més, pertànyer a més d'una classe d'objecte, per exemple, un individu és definit per la classe persona, però també pot ser definit per atributs de les classes `inetOrgPerson`, `groupOfNames` i `organization`.

L'estructura d'objectes del servidor (anomenat *schema*) determina quins són els atributs permesos per a un objecte d'una classe (que es defineixen en el fitxer `/etc/ldap/schema` com `inetorgperson.schema, nis.schema, opeldap.schema, corba.schema`, etc.). Totes les dades es representen com un parell atribut = valor, on l'atribut és descriptiu de la informació que conté; per exemple, l'atribut utilitzat per a emmagatzemar el nom d'una persona és `commonName`, o cn, és a dir, una persona anomenada Remix Nteum es representarà per cn: Remix Nteum i portarà associat altres atributs de la classe persona com `givenname: Remix surname: Nteum mail: nteum@uoc.edu`. A les classes hi ha atributs obligatoris i optatius i cada atribut té una sintaxi associada que indica quin tipus d'informació conté l'atribut, per exemple, `bin` (*binary*), `ces` (*case exact string*, ha de buscar-se igual), `cis` (*case ignore string*, poden ignorar-se majúscules i minúscules durant la cerca), `tel` (*telephone number string*, s'ignoren espais i '-') i `dn` (*distinguished name*). Un exemple d'un arxiu en format LDIF podria ser:

```
dn: dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit


dn: ou = groups, dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit
ou: groups


dn: ou = people, dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit
ou: people


dn: cn = Remix Nteum, ou = people, dc = UOC, dc = com
cn: Remix Nteum
sn: Nteum
objectclass: top
objectclass: person
objectclass: posixAccount
objectclass: shadowAccount
   uid:remix userpassword:{crypt}p1pss2ii(0pgbs*do& = )eksd uidnumber:104
   gidnumber:100
   gecos:Remix Nteum
   loginShell:/bin/bash
   homeDirectory: /home/remix
   shadowLastChange:12898
   shadowMin: 0
   shadowMax: 999999
   shadowWarning: 7
   shadowInactive: -1
   shadowExpire: -1
   shadowFlag: 0


dn:
```

```
cn = unixgroup, ou = groups, dc = UOC, dc = com
objectclass: top
objectclass: posixGroup
cn: unixgroup
   gidnumber: 200
   memberuid: remix
   memberuid: altre-usuari
```

Les línies llargues es poden continuar a sota començant per un espai o un tabulador (format LDIF). En aquest cas, s'ha definit la base DN per a la institució `dc = UOC, dc = com`, la qual conté dues subunitats: `people` i `groups`. A continuació, s'ha descrit un usuari que pertany a `people` i a `group`. Una vegada preparat l'arxiu amb les dades, aquest ha de ser importat al servidor perquè estigui disponible per als clients LDAP. Hi ha eines per a transferir dades de diferents bases de dades a format LDIF [19]. Sobre Debian, s'ha d'instal·lar el paquet `slapd` i `ldap-utils`, que és el servidor d'OpenLdap i un conjunt d'utilitats per a accedir a servidors locals i remots. Durant la instal·lació, sol·licitarà un *passwd* per a gestionar l'administració del servei, es generarà una configuració inicial i s'engegarà el servei que es pot verificar amb l'ordre `slapcat` que, prenent la informació disponible (FQDN de */etc/hosts*), generarà una cosa similar a aquesta:

```
   dn: dc=nteum,dc=org
   objectClass: top
   objectClass: dcObject
   objectClass: organization
   o: nteum.org
   dc: nteum
...

   dn: cn=admin,dc=nteum,dc=org
   objectClass: simpleSecurityObject
   objectClass: organizationalRole
   cn: admin
   description: LDAP administrator     userPassword:: e1NTSEF9TU9jVE1qWllPVFBmd2FiZWJtSjcrY0pYd2wvaTk5aUc=
...
```

Amb les ordres `ldapadd` i `ldapdelete`, es poden agregar/esborrar registres de la taula del servei utilitzant normalment arxius de text (sobretot per l'`add`) on estiguin definits els nous registres. Si bé no és un procediment complex per a gestionar el servidor, pot ser més adequat, en els primers passos, utilitzar algunes de les aplicacions que permeten gestionar el servidor d'una manera més amigable com per exemple `phpldapadmin` (administració per mitjà d'apache+php), `jxplorer` (aplicació en java), `gosa` o `lat` (tots aquests en la majoria de distribucions). En el nostre cas instal·larem `phpldapadmin`, que combina simplicitat i permet gestionar la majoria d'opcions del servidor Ldap.

Per a instal·lar-lo, fem `apt-get install phpldapadmin`, el qual ja reiniciarà el servidor Apache però, si no, fa que es reinicïi `/etc/init.d/apache2 restart`. Per a connectar-nos a un navegador, introduïm la direcció web: `http://localhost/phpldapadmin/`. Sortirà la pàgina de benvinguda i en la banda esquerra podrem fer el *login* (si el *ServerName* d'Apache no coincideix amb el LDAP, no demanarà com a usuari DN, per la qual cosa li haurem d'indicar el correcte cn=admin, dc=nteum, dc=org i la contrasenya del servi-

dor LDAP). Una vegada connectat al servidor, seleccionem l'arrel (dc=nteum, dc=org) i seleccionem entre els *templates* que apareixen una *Organizational Unit (OU)* que anomenarem `users`, repetim els passos (opció `create new entry` des de l'arrel) i creem una altra *OU* que anomenem `groups`. Ara només ens queda crear els usuaris i els grups i assignar els usuaris als seus grups. Dins de la unitat organitzativa `groups`, crearem els grups `vendes` (gid=1001) i `compres` (gid=1002) i en la unitat organitzativa `users` crearem els usuaris `juan pirulo` (uid=1001, vendes, id=jpirulo) i `ana pirulo` (uid=1002, compres, id=apirulo). Per als grups, seleccionem l'OU `groups`, fem `create new child` i seleccionem `Posix Group`. Crearem els dos grups indicats però haurem de modificar el `gid` ja que els assigna per defecte a partir de 1.000 i nosaltres els volem diferents. Repetim l'operació en `users` seleccionant `User Account`. Aquí ens demanarà les dades dels usuaris (nom, cognom, grup, contrasenyes, *home directory*, *shell*, etc.) i després haurem de canviar el `uid`, ja que els assigna per defecte. Després, podem observar novament el contingut executant `slapcat` en les noves dades generades. Phpldapadmin es pot configurar amb l'arxiu */usr/share/phpldapadmin/config/config.php*, però en la majoria de casos la configuració per defecte ja és operativa (en alguns casos, pot ser necessari adaptar la línia $servers->setValue('server','base',array('dc=nteum,dc=org')); per a adequar a les dades del nostre servidor).

Per a configurar un client, haurem d'instal·lar els paquets `slapd` i `ldap-utils` amb `apt-get install slapd ldap-utils`, que ens sol·licitarà una sèrie d'informació: URI del servidor (ldap://192.168.1.33), el DC (dc=nteum,dc=org), la versió (3), l'usuari administratiu (cn=admin,dc=nteum,dc=org) i el *passwd*. Ens informarà que fem el canvi manual de nsswitch (ok), permetem a PAM canviar els *passwords* locals (Yes), *enforces login* (No), *admin acount suffix* (cd=admin,dc=nteum,dc=org), i finalment el *passwd* novament.

A continuació, hem de modificar l'arxiu */etc/nsswitch.conf* amb:

```
passwd: compat ldap
group: compat ldap
shadow: compat ldap
...
netgroup: ldap
```

Finalment haurem de modificar l'arxiu */etc/pam.d/common-password* (traient de la línia l"use_authtok')

```
password [success=1 user_unknown=ignori default=die] pam_ldap.so
try_first_pass
```

i en */etc/pam.d/common-session* agregar al final (per a crear el *home directory* de manera automàtica)

```
session optional pam_mkhomedir.so skel=/etc/skel umask=077.
```

Després, haurem de reiniciar la màquina (`shutdown -r now`) i ja ens podrem connectar amb els usuaris que hem creat (jpirulo o apirulo) des de la interfície gràfica.

1.13. Serveis d'arxius (NFS, *Network File System*)

El sistema NFS permet a un servidor exportar un sistema d'arxiu perquè pugui ser utilitzat de manera interactiva des d'un client. El servei es compon bàsicament d'un servidor (bàsicament representat per `nfsd*`) i un client (representat per `rpc.mountd`) que permeten compartir un sistema d'arxiu (o part d'aquest) a través de la xarxa. En l'última versió de NFSv4, s'inclouen una sèrie de *daemons* més com *idmapd*, *statd*, a més d'una sèrie de mòduls per a les noves funcionalitats d'aquesta versió. En Debian, instal·leu `apt-get install nfs-common` (serà necessari el paquet *rpcbin* que, com es va comentar abans, és el nou *portmap* i generalment ja està instal·lat) per al client, mentre que el servidor necessita `apt-get install nfs-kernel-server`. El servidor (en Debian) es gestiona mitjançant l'*script* `/etc/init.d/nfs-kernel-server` o simplement amb `service nfs-kernel-server start|stop`. El servidor fa servir un arxiu (`/etc/exports`) per a gestionar l'accés remot als sistemes d'arxiu i el seu control. Sobre el client (o un altre usuari mitjançant `sudo`), el *root* pot muntar el sistema remot a través de l'ordre:

```
mount -t nfs Ipserver:directori-remot directori_local
```

i a partir d'aquest moment, el `directori-remot` es veurà dins del directori local (aquest ha d'existir abans d'executar el `mount`). Aquesta tasca en el client es pot automatitzar utilitzant l'arxiu de *mount* automàtic (`/etc/fstab`) incloent una línia; per exemple:

```
remix.world:/home /home nfs defaults 0 0.
```

Això ens indica que es muntarà el directori `/home` del *host* `remix.world` en el directori local `/home`. A més, aquest sistema d'arxiu es muntarà amb els paràmetres per defecte (vegeu `man mount`, apartat `mount options for ntfs` i `man nfs` per a opcions específiques per a NFSv4). Els últims dos zeros indiquen que el sistema d'arxius no ha de ser *dumped* i que no s'hi activarà el *fsck*. L'arxiu */etc/exports* serveix d'ACL (llista de control d'accés) dels sistemes d'arxiu que poden ser exportats als clients. Cada línia conté un sistema de fitxers (*filesystem*) per a exportar seguit dels clients que el poden muntar, separats per espais en blanc. A cada client se li pot associar un conjunt d'opcions per a modificar el comportament (consulteu *man exports* per a veure un llista detallada de les opcions). Un exemple d'això podria ser:

```
# Exemple de /etc/exports
/           master(rw) trusty(rw,no_root_squash)
/projects   proj*.local.domain(rw)
/usr        .local.domain(ro) @trusted(rw)
/pub        (ro,insecure,all_squash)
/home       195.12.32.2(rw,no_root_squash) www.first.com(ro)
/user       195.12.32.2/24(ro,insecure)
/home       192.168.1.0/24(rw,sync,fsid=0,no_root_squash,no_subtree_check)
```

La primera línia exporta el sistema d'arxius sencer (`/`) a `master` i `trusty` en mode lectura/escriptura. A més, per a `trusty` no hi ha *uid squashing* (el *root* del client accedirà com a *root* als arxius *root* del servidor, és a dir, els dos *root* són equivalents malgrat ser de màquines diferents i s'utilitzen per a màquines sense disc). La segona i tercera línies mostren exemples de '*' i de *netgroups* (indicats per `@`). La quarta línia exporta el directori `/pub` a qualsevol màquina del món, només de lectura, i permet l'accés de clients NFS que no utilitzen un port reservat pel NFS (opció `insecure`) i tot s'executa sota l'usuari *nobody* (opció `all esquaix`). La cinquena línia especifica un client per a la seva IP i en la sisena, s'especifica el mateix però amb màscara de xarxa (/24) i amb opcions entre parèntesis sense espai de separació. Només hi pot haver espai entre els clients habilitats. L'última línia exporta el directori */home* a totes les màquines de la xarxa 192.168.0.* en mode sincronitzat, de lectura/escriptura, amb accés del *root* remot, fsid és la identificació de sistema d'arxiu i no_subtree_check indica que no es farà la verificació de la ruta/arxiu en una petició sobre el servidor.

Dues ordres útils per a treballar amb l'nfs són l'`exportfs` (que mostra i ens permet actualitzar les modificacions que s'hagin fet sobre l'arxiu */etc/exports*) i `nfsiostat/nfsstat`, que ens permetrà obtenir estadístiques de funcionament sobre NFS i observar el seu funcionament.

1.14. Servidor de wiki

Un (o una) **wiki** (del hawaià *wiki wiki*, "ràpid") és un lloc web col·laboratiu que pot ser editat per diversos usuaris que poden crear, editar, esborrar o modificar el contingut d'una pàgina web, de manera interactiva, fàcil i ràpida; aquestes facilitats fan d'una wiki una eina eficaç per a l'escriptura col·laborativa. La tecnologia wiki permet que pàgines web allotjades en un servidor públic (les pàgines wiki) siguin escrites de manera col·laborativa mitjançant un navegador, utilitzant una notació senzilla per a donar format, crear enllaços, etc. i conservant un historial de canvis que permet recuperar de manera senzilla qualsevol estat anterior de la pàgina. Quan algú edita una pàgina wiki, els canvis apareixen immediatament a la web, sense passar per cap tipus de revisió prèvia. *Wiki* també es pot referir a una col·lecció de pàgines d'hipertext, que qualsevol persona pot visitar i editar (definició de Wikipedia). Debian té el seu wiki

Enllaç d'interès

Per a saber més sobre MoinMoin, podeu visitar la seva pàgina web en: http://moinmo.in.
En concret, trobareu les instruccions detallades per a instal·lar MoinMoin en: http://master19.moinmo.in/InstallDocs.

en http://wiki.debian.org/ o també Apache en http://wiki.apache.org/general/ i ambdues estan basades en **MoinMoin**. MoinMoin és una *Python WikiClone* que permet inicialitzar ràpidament la seva pròpia wiki i només es necessiten un servidor de web i el llenguatge Python instal·lat. A la web de MoinMoin, es troben les instruccions detallades per a instal·lar MoinMoin, però hi ha dues maneres principals de fer-ho: instal·lació ràpida i instal·lació de servidor.

1.14.1. Instal·lació ràpida

1) Descarregar el paquet des de http://moinmo.in/moinmoindownload que serà, per exemple, per a la versió 1.9 `moin-1.9.7.tar.gz`. Si es vol verificar la integritat del paquet, es pot fer `md5sum moin-x.x.x.tar.gz` i verificar que coincideixin el *hash* generat amb aquell que hi ha a la pàgina de descàrrega.

2) Desempaquetar MoinMoin `tar xvzf moin-x.x.x.tar.gz`. Això crearà un directori `moin-x.x.x` en el directori actual amb els arxius en el seu interior.

3) Com que MoinMoin està escrita en Python, és necessari utilitzar l'intèrpret de Python:

```
cd moin-x.x.x; python wikiserver.py
```

Aquesta ordre mostrarà per pantalla els missatges d'execució del servidor. Entre aquesta informació es mostrarà l'adreça IP sobre la qual està corrent el servidor, que podrà ser alguna cosa com `http://127.0.0.1:8080`. Aquesta opció fa servir un servidor web intern, serà accessible des de la direcció `http://localhost:8080/` i funcionarà fins que es pressioni `Ctrl-C` en el terminal.

1.14.2. Instal·lació de servidor

MoinMoin és una aplicació WSGI (*Web Server Gateway Interface*) i, per tant, el millor entorn per a executar Moin Moin és un que permeti WSGI com, per exemple, Apache amb mod_wsgi. En Debian, podem instal·lar el mòdul instal·lant `apt-get install libapache2-mod-wsgi`.

Enllaç d'interès

Les instruccions per a instal·lar WSGI per a Apache i configurar MoinMoin en aquest cas es poden trobar en la següent adreça: http://moinmo.in/HowTo/ApacheWithModWSGI.

Instal·lació de MoinMoin

Per a instal·lar MoinMoin, s'ha de descarregar l'última versió i descompactar l'arxiu (per exemple, `tar xvzf moin-1.9.7.tar.gz`) i després fer una `cd moin-1.9.7/` i, a continuació, executar:

```
python setup.py install --force --record=install.log --prefix='/usr/local'
```

Per a fer un test simple:

```
cd /usr/local/share/moin/server
python test.wsgi
```

En el navegador, introduir com a URL localhost:8000 i veurem la pàgina de test de WSGI.

Copiar la configuració:

```
cd /usr/local/share/moin
cp server/moin.wsgi .
cp config/wikiconfig.py .
```

Agregar un arxiu en */etc/apache2/conf.d/moin.conf* amb el següent contingut:

```
# MoinMoin WSGI configuration
# you will invoke your moin wiki at the root url, like http://servername/FrontPage:
WSGIScriptAlias / /usr/local/share/moin/moin.wsgi
# create some wsgi daemons - use these parameters for a simple setup
WSGIDaemonProcess moin user=www-data group=www-data processes=5
threads=10
maximum-requests=1000 umask=0007
# use the daemons we defined above to process requests!
WSGIProcessGroup moin
```

Modificar l'arxiu */usr/local/share/moin/moin.wsgi* agregant al final del paràgraf a2: sys.path.insert(0, '/usr/local/share/moin')

Modificar els permisos dels directoris/pàgines:

```
cd /usr/local/share; chown -R www-data:www-data moin;
chmod -R ug+rwX moin; chmod -R o-rwx moin
```

Verificar que tinguem un *site* per defecte en Apache (si no, s'ha de fer `a2ensite default`) i reiniciar Apache (`service apache2 restart`)

Si ens connectem a l'URL localhost, tindrem la pàgina inicial de MoinMoin. Per a configurar el nom de la wiki i l'usuari administrador, podem editar l'arxiu */usr/local/share/moin/wikiconfig.py*, traiem el comentari de `page_front_page = o"FrontPage"` i indiquem el nom de l'administrador, p. ex., `superuser = [o"WikiAdmin", ]`, i reiniciem Apache novament. Per a configurar el llenguatge, hem d'entrar com a administrador (WikiAdmin) (si no tenim un usuari, seleccionem *login*, seleccionem 'you can create one now' i i el creem; ha de coincidir amb aquell que introduïm com a superuser). Després, podrem configurar l'idioma des de

```
http://localhost/LanguageSetup?action=language_setup
```

i a partir d'aquesta acció, ja podrem començar a crear la nostra primera wiki (informació addicional en http://moinmo.in/howto i particularment en l'adreça http://moinmo.in/howto/ubuntuquick).

Per a configurar múltiples wikis, primer heu de copiar `config/wikifarm/*` de la distribució en el directori `moin/config/`. Després, s'han de seguir les

instruccions anteriors per a cadascuna de les wikis de la col·lecció (*farm*), tenint en compte que:

1) és necessari tenir `data_dir` i `data_underlay_dir` separats per a cada wiki,

2) si busqueu que comparteixin alguna configuració, llavors aquesta ha d'estar en `farmconfig.py` i les específiques han d'estar en `mywiki.py`.

1.15. Gestió de còpies de seguretat (*backups*)

Les còpies de seguretat (*backup*) es refereixen a una còpia de les dades originals que es fa amb la finalitat de disposar d'un mitjà per a recuperar-les en cas de pèrdua total o parcial a causa de fallades en els dispositius físics, esborrats per accident o atacs informàtics, infectats per virus o altres causes que fan que la informació no existeixi o no sigui la desitjada. El procés de còpia de seguretat es complementa amb un procés de restauració de les dades (*restore*) que pot ser total o parcial/selectiu, que permet retornar el sistema informàtic al punt en el qual es van emmagatzemar les dades. Això pot significar la pèrdua d'informació entre el moment en què es fa la còpia de seguretat i el moment en què es detecta que les dades no existeixen o estan corrompudes, per la qual cosa la política de planificació de còpies de seguretat ha de ser una de les activitats importants en tot administrador de sistemes.

S'ha de tenir en compte que la pèrdua de dades és habitual (i no per això sense conseqüències i, en alguns casos, fatals), ja que, d'acord amb estadístiques recents, més del 60% dels usuaris d'Internet declaren haver patit una seriosa pèrdua de dades en alguna ocasió, i segons un estudi de la Universitat de Texas, només el 6% d'empreses amb pèrdua catastròfica de dades sobreviurà, enfront d'un 43% que mai reobrirà el negoci i un 51% que haurà de tancar en un termini de 2 anys. Per a reafirmar més encara la necessitat de còpies de seguretat, i pel que fa a aquells sistemes que continguin dades de caràcter personal i que estiguin subjectes a la legislació del país (p. ex., a l'Estat espanyol la LOPD, Llei Orgànica de Protecció de Dades), una de les obligacions que han de complir les empreses/institucions/individus és tenir còpies de seguretat per a preservar les dades que tenen emmagatzemades i que estan subjectes a aquesta normativa.

1.15.1. Programes habituals de còpies de seguretat

Hi ha diverses opcions per a fer còpies de seguretat amb diferents objectius, prestacions i interfícies en totes les distribucions GNU/Linux (p. ex., `backintime`, `bacula`, `backup2l`, `backuppc`, `bup`, `chiark`, `dejadup`, `dirvish`, `flexbackup`, `lucky`, `rdiff`, `vbackup`, entre d'altres). Una de les més potents és **Bacula*** (http://blog.bacula.org/), que és una col·lecció d'ei-

***http://www.bacula.org**

nes per a fer còpies de seguretat en una xarxa. Bacula es basa en una arquitectura client/servidor que resulta molt eficaç i fàcil de manejar, ja que presenta un conjunt molt ampli de característiques i és eficient tant per a un conjunt d'ordinadors personals com per a grans instal·lacions. El paquet està format per diferents components, entre els més importants es poden trobar:

- **Bacula-director**, *daemon* que gestiona la lògica dels processos de *backup*.
- **Bacula-storage**, *daemon* encarregat de manejar els dispositius d'emmagatzematge.
- **Bacula-file**, *daemon* per mitjà del qual Bacula obté els fitxers que necessita per a fer la còpia de seguretat i que s'hauran d'instal·lar en les màquines font dels fitxers que cal fer còpia de seguretat, i
- **Bacula-console**, que permet interactuar amb el servei de *backup*.

Bacula suporta discs durs, cintes, DVD, USB i també diferents bases de dades (MySQL, PostgreSQL i SQLite), però com a contrapartida és necessari disposar de tots els paquets instal·lats i la seva instal·lació i posada a punt poden resultar complexes.

Un altre paquet interessant és **BackupPC***, que permet fer còpies de seguretat de disc a disc amb una interfície basada en el web. El servidor s'executa en qualsevol sistema Gnu/Linux i admet diferents protocols perquè els clients puguin escollir la forma de connectar-se al servidor. Aquest programa no és adequat com a sistema de còpia de seguretat d'imatges de disc o particions, ja que no suporta còpies de seguretat en un nivell de bloc de disc; no obstant això, és molt simple de configurar i la possible intrusió sobre la xarxa d'ordinadors en la qual es desitja fer el suport és mínima. Aquest servidor incorpora un client *Server Message Block* (SMB) que es pot utilitzar per a fer còpia de seguretat de recursos compartits de xarxa d'equips que executen Windows.

***http://backuppc.sourceforge.net/info.html**

La seva instal·lació és senzilla fent, en Debian, `apt-get install backuppc`. Ens indicarà que seleccionem el servidor web (apache2) i ens indicarà l'usuari i *passwd* que ha creat, no obstant això, aquests *passwd*/usuari es poden canviar amb `htpasswd /etc/backuppc/htpasswd backuppc`. Després, en el nostre navegador posem com a URL http://localhost/backuppc i amb l'usuari/*passwd* accedirem a la pàgina principal de l'aplicació on ens donarà l'estat del servidor i les opcions.

Hi ha diverses formes de configurar els clients per a fer les còpies de seguretat i dependrà del mètode per a fer-ho i del sistema operatiu. Per a això, cal consultar en l'apartat de documentació com es configurarà cadascuna de les transferències (http://backuppc.sourceforge.net/faq/BackupPC.html).

En el cas d'un sistema (remot) Gnu/Linux, des de la interfície d'administració cal editar-ne la configuració (*host* i *xfer*) i confirmar que s'ha definit com a mètode `rsync` i el directori per a fer la còpia. Com que les còpies de seguretat es fan mitjançant `rsync` en combinació amb `ssh`, és necessari que l'usuari

backuppc del servidor pugui accedir com a *root* sense clau a la màquina remota. Per a això, s'ha d'adoptar l'entitat de l'usuari *backuppc* (`su - backuppc`) i generar les claus (sense *passwd*) `ssh-keygen -t dsa` i després utilitzar l'ordre `ssh-copy-id root@client` per a copiar la clau. S'ha de verificar que es pot accedir a l'usuari *root* del client a través de `ssh` i sense *passwd*. A partir d'aquest punt, n'hi haurà prou de seleccionar l'equip remot en què s'ha de fer el *backup* des de la interfície d'administració i iniciar una primera còpia seleccionant el botó *Començar* còpia de seguretat completa.

En sistemes Windows, la manera més simple de portar a terme *backups* és mitjançant el protocol SMB, per la qual cosa sobre el sistema Windows s'haurà d'ingressar com a administrador i configurar el sistema per a compartir carpetes de les quals es vulgui fer la còpia de seguretat o bé el disc dur complet (per exemple, C:), i definir un usuari/*passwd* per a compartir aquest recurs. Des de la interfície d'administració, s'ha d'editar la configuració del host remot amb Windows del qual s'han de fer les còpies de seguretat (per la seva IP, per exemple), l'usuari i el mètode smb (no oblideu fer Save després d'haver modificat aquestes dades). Definiu en la pestanya *Xfer* el nom del recurs compartit que cal fer les còpies de seguretat en l'equip remot i el nom de l'usuari i clau d'accés de recurs compartit de l'equip Windows remot. A partir d'aquest punt, n'hi haurà prou de seleccionar l'equip remot des de la interfície d'administració i iniciar un primer suport seleccionant el botó Començar còpia de seguretat completa.

Amb Backuppc, es pot definir la freqüència dels suports totals i suports incrementals. De manera predeterminada, el valor per als suports totals és cada 7 dies i per als incrementals és cada dia. Es recomana utilitzar un valor lleugerament inferior als dies. Per exemple, 6,97 en lloc de 7 dies i 0,97 en lloc d'1 dia per a millorar la granularitat del suport (recordeu sempre fer *Save* després de modificar cada opció).

1.15.2. rdiff-backup i rdiff-backups-fs

Per a administradors o usuaris avançats, hi ha l'opció de `rediff-backup`, que és una ordre per a la còpia de seguretat d'un directori a un altre i que també pot ser a través d'una xarxa. El directori de destinació posseirà una còpia del directori font però també informació addicional (diffs) per a gestionar millor les còpies incrementals (encara d'arxius antics). L'aplicació també preserva subdirectoris, enllaços no simbòlics (*hard links*), arxius dev, permisos, propietat (uid/gid), dates de modificació, atributs estesos i ACL i pot operar de manera eficient a través d'un *pipe* a *rsync* per exemple, o utilitzar `rdiff-backup + ssh` per a fer *backups* incrementals en lloc remot transmetent només les diferències. També és habitual (i molt simple) tenir un proveïdor de serveis (Gdrive, Dropbox, etc.) amb un directori sobre l'ordinador local que serà sincronitzat sobre el *cloud* del proveïdor i fer la còpia `rdiff-backup` sobre aquest directori perquè després es transmeti al *cloud* del proveïdor. La forma habitual de treball (després d'instal·lat l'ordre) és molt simple: `rdiff-backup font`

`destinació` i en el cas remot /algun/dir-local a /algun/dir-remot sobre la màquina hostname.org serà

```
rdiff-backup /algun/dir-local hostname.org::/algun/dir-remot
```

però pot ser al revés

```
rdiff-backup user@hostname.org::/remote-dir local-dir
```

i també podrien ser sobre dues màquines remotes

```
rdiff-backup -v5 -print-statistics user1@host1::/source-dir user2@host2::/dest-dir.
```

Per a recuperar un directori local, es fa simplement mitjançant la còpia i si és remot, `rdiff-backup -r now hostname.org::/remote-dir/file local-dir/file` (podeu veure uns quants exemples més en la següent adreça web: http://www.nongnu.org/rdiff-backup/examples.html).

Un dels problemes de recuperar la informació prèviament guardada és que accedir als arxius de còpia de seguretat més recent és fàcil (solament s'ha d'introduir el directori de còpia de seguretat), però és complicat si desitgem accedir i navegar a través de les versions anteriors. `rdiff-backup` permet accedir-hi per un conjunt d'instruccions, que requereixen un coneixement precís dels noms d'arxiu i els temps de còpia de seguretat i que pot ser complicat de recordar o seleccionar. `rdiff-backup-fs` (https://code.google.com/p/rdiff-backup-fs/) permet crear un sistema d'arxius en espai d'usuari basat en la informació de rdiff. Per a això s'utilitza FUSE (*Filesystem in userspace**), que permet que qualsevol usuari pugui muntar el sistema d'arxius guardat per rdiff i navegar per cada increment i còpia efectuada.

***http://fuse.sourceforge.net**

Una alternativa per a les còpies de seguretat d'un sistema remot per a ssh és utilitzar `sshfs` (http://fuse.sourceforge.net/sshfs.html), que permet l'accés a l'espai d'usuari a un directori remot mostrant-lo localment, per la qual cosa després aplicant rdiff és possible fer còpies incrementals d'aquest recurs remot.

1.16. *Public Key Infrastructure* (PKI)

Per PKI (*Public Key Infrastructure*) s'entén un conjunt de maquinari i programari, polítiques i procediments de seguretat que permeten l'execució amb garanties d'operacions com el xifratge, la signatura digital o el no-repudi de transaccions electròniques. El terme PKI s'utilitza per a referir-se tant a l'autoritat de certificació com a la resta de components, si bé de vegades, de manera errònia, s'utilitza per a referir-se a l'ús d'algorismes de clau pública. En una operació en què s'utilitzi PKI intervenen, a més de qui inicia l'acció i el seu destinatari, un conjunt de serveis que donen validesa a l'operació i garanteixen que els certificats implicats són vàlids. Entre aquests podem comptar amb l'autoritat

de certificació, l'autoritat de registre i el sistema de segellament de temps. La infraestructura PKI utilitza procediments basats en operacions criptogràfiques de clau pública, amb algorismes de xifratge ben coneguts, accessibles i segurs i és per això que la seguretat està fortament lligada a la privadesa de la clau privada i les polítiques de seguretat aplicades per a protegir-la. Els principals usos de la PKI són, entre d'altres els següents: autenticació d'usuaris i sistemes (*login*), identificació, xifratge, signatura digital, comunicacions segures i garantia de no-repudi.[21][22][24]

Com s'ha vist anteriorment (per a Apache+SSL), la generació de certificats digitals és extremadament necessària dins de les necessitats habituals dels administradors i usuaris dels sistemes d'informació, per exemple, per a accedir a una pàgina SSL o per a signar un correu electrònic. Aquests certificats es poden obtenir de les entitats certificadores privades com per exemple StartComm (https://www.startssl.com/) i Verisign (http://www.verisign.es/), que tenen alguns productes gratuïts (per exemple per a signar correus), però, en la majoria, els productes tenen un cost elevat. També podem recórrer a utilitzar GNUPG (https://www.gnupg.org/), que és *Open-Source* i per a determinats finalitats és correcte però no per a altres, o bé considerar entitats de certificació institucionals, per exemple a Catalunya IdCat (http://idcat.cat/), que ens permetran obtenir certificats de ciutadà (gratuïts) per a signatura, encriptació, autenticació, no repudi però no per a servidors (Idcat sí que pot expedir un altre tipus de certificats però només és per a l'Administració pública i universitats del país). En el present apartat, instal·larem i configurarem una entitat certificadora arrel (i subentitats CA per als diferents dominis de control d'aquesta CA) basada en l'aplicació TinyCA (si bé existeixen altres paquets com OpenCA –https://pki.openca.org/– que són més potents i escalables però són més complexos en la seva configuració) que s'adapta molt bé per a petites i mitjanes institucions/empreses. Una entitat certificadora és la base de la infraestructura PKI i emet certificats per a donar garanties d'autenticitat d'una comunicació, un lloc o una informació. Quan instal·lem Apache, hem autosignat el certificat digital (és a dir, nosaltres hem fet tots els passos: petició, generació i signatura) que codifica la comunicació però això no dóna garantia si no es verifica la signatura del certificat autosignat. Per a solucionar aquest problema, i sense recórrer a un proveïdor públic/privat, crearem la nostra pròpia estructura de CA i distribuirem per canals segurs als nostres usuaris/clients el certificat personal / de servidors i el certificat arrel de la CA perquè els instal·lin en els seus navegadors (com ja ho estan els de StartComm, Verisign, Catcert/Idcat o altres entitats de certificació). Això es fa de manera que quan un lloc web, per exemple, presenti al navegador un certificat digital per a codificar la comunicació SSL, el navegador reconegui el lloc pel certificat arrel que té instal·lat i hi confiï (i no surti la típica finestra d'advertiment de lloc insegur) mantenint la privadesa de les comunicacions. Els usos d'aquests certificats creats per aquesta CA poden ser diversos: per a signar/encriptar un correu, per a validar els nostres servidors SSL o per a configurar la VPN, entre d'altres.

La pràctica habitual és tenir una CA i crear Sub-CA (una per a cada domini) perquè el sistema sigui escalable, per la qual cosa la CA signarà la creació de

noves Sub-CA i després d'aquestes (el seu responsable), tindran capacitat per a signar els seus certificats i totes tindran el mateix certificat arrel de la root-CA (guia molt detallada en [25]). Una vegada instal·lada (`apt-get install tinyca`), executem `tinyca2` i ens presentarà la pantalla principal i indicacions per a crear una nova CA que serà la rootCA. Completem la pantalla amb les dades, p. ex., *Name=Rootca-nteum.org, Data for CA=Rootca-nteum.org, SP, passwd, BCN, BCN, NTEUM, NTEUM, adminp@sysdw.nteum.org, 7300, 4096, SHA-1* per a tots els camps. Quan fem OK, apareixerà una nova pantalla per a configurar la rootCA. Cal seleccionar *"Certificate Signing/CRL Signing", "critical"* i en Netscape *Certificate Type="SSL CA, S/MIME CA, Object Signing CA."*, i si es desitja posar un URL per a revocar els certificats (la qual cosa pot ser una bona idea) es poden emplenar els camps corresponents, després finalment OK. Amb això es crearan els certificats i apareixerà la pantalla principal de tinyCA amb 4 tabuladors on indica *CA* (informació sobre la CA activa), *Certificates* (mostra els certificats creats per la CA) *Keys* (mostra les claus dels certificats) i *Requests* (peticions de certificats que esperen ser signats per la CA). Per damunt, apareixen una sèrie d'icones per a accedir a funcions directes però Tinyca2 té un error i no mostra els noms de les icones.

El següent pas és crear una nova sub-CA i, per a fer-ho, verificar que estem en el tab de la CA i fer clic en la tercera icona des de la dreta, que ens mostrarà una finestra que com a subtítol té "*Create a new Sub CA*". Haurem d'introduir el *passwd* que vam posar en la rootCA, donar-hi un nom (subca-sysdw.nteum.org), *Data-for-CA* (sysdw.nteum.org) i la resta de dades com vam fer anteriorment (el *passwd* no ha de ser necessàriament el mateix de la root-CA) i quan fem OK ens sortirà la finestra principal però amb la Sub-CA seleccionada, si volem tornar a la CA haurem d'anar al menu *File->Open* i obrir la rootCA. Recordeu que la Sub-CA serà qui gestionarà les peticions i signarà els certificats per a aquest domini, per la qual cosa hem de seleccionar l'adequada a cada moment. La rootCA només la utilitzarem per a crear/revocar noves sub-CA i per a exportar el certificat arrel de la rootCA (necessari per a enviar-los als nostres clients perquè l'instal·lin en els seus navegadors).

Per a crear un certificat que permeti certificar el nostre *web-server* quan utilitza SSL (https://sysdw.nteum.org), amb la nostra Sub-CA seleccionada anem al tab de *Request* i amb el botó dret seleccionem "*New request*" i s'obrirà una finestra on haurem d'introduir l'URL del nostre servidor (sysdw.nteum.org) com a *CommonName* i emplenar la resta de dades. Una vegada creat, el seleccionem i amb el botó dret indiquem "*Sign request*" i seleccionem "*Server request*", que ens mostrarà una finestra amb el *password* de la CA i la validesa. Aquest procediment és el que genera confiança ja que estem validant la informació aportada en la petició i signant el certificat (que ja podrem veure en els tabs corresponents).

Ara haurem d'exportar els certificats (del web i la rootCA), la *key* i configurar Apache perquè els incorpori. En primer lloc, exportarem la rootCA i la introduirem en Firefox/Iceweasel. Per a això, en la finestra principal *File->OpenCA* seleccionem la nostra rootCA i seleccionant la segona icona de la

dreta que correspon a "*Export CA Certificate*" indiquem un nom d'arxiu i el format (PEM és l'adequat) i desem el certificat en el nostre sistema d'arxiu (p. ex., /tmp/Rootca-nteum.org.pem). És important tenir en compte fer un `chmod 644 /tmp/Rootca-nteum.org.pem`, ja que es desarà com a 600 i altres usuaris no el podran importar. Des de Firefox/Iceweasel seleccionem *Preferences/Setting-> Advanced-> Certificates-> View Certificates-> Authorities-> Import* i seleccionem l'arxiu abans creat marcant totes les *trust settings* que ens presenta en la finestra següent (3). Després, podrem veure amb el nom que vam donar a *Organization* el certificat corresponent.

A continuació, en TinyCA2 obrim la nostra sub-CA, seleccionem el tab *Certificates* i sobre el certificat seleccionem el botó dret i indiquem *Export certificate* i donem un nom d'arxiu, per exemple sysdw.nteum.org-cert.pem, després repetim el procés amb la *key* en el *Key* tab, fem *Export key* i el desem, per exemple com a sysdw.nteum.org-key.pem. És important decidir si posem *passwd* o no, ja que si el posem cada vegada que arrenqui el servidor ens sol·licitarà el *passwd*. Sobre el tab CA haurem d'exportar ara el *certificate chain*, que és la primera icona des de la dreta i desar-lo com a sysdw.nteum.org-chain.pem. Mourem aquests tres arxius a */etc/ssl/private/* i passarem a configurar Apache modificant l'arxiu que configuri el nostre lloc SSL, per exemple nosaltres hem utilitzat */etc/apache2/sites-available/default-ssl*, en el qual hem modificat (només es mostra les línies principals):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerName sysdw.nteum.org
...
    SSLEngine on
    SSLCertificateFile /etc/ssl/private/sysdw.nteum.org-cert.pem
    SSLCertificateKeyFile /etc/ssl/private/sysdw.nteum.org-key.pem
    SSLCertificateChainFile /etc/ssl/private/sysdw.nteum.org-chain.pem
...
</VirtualHost>
</IfModule>
```

Només ens queda habilitar el lloc (`a2ensite default-ssl`) i reiniciar el servidor Apache (`service apatxe2 restart`) (ens demanarà el *passwd* si l'hem posat en la *key*) i provar (en el navegador en què tenim instal·lat el certificat arrel de la rootCA) l'URL https://sysdw.nteum.org. Si tot està correcte, carregarà la pàgina sense la típica finestra que informa que el lloc no és segur.[25]

Per a crear certificats per a una adreça de correu i distribuir-los als nostres usuaris juntament amb el certificat de la rootCA, podem fer en el tab *Certificates* de la nostra sub-CA, seleccionar *New - Create Key and Certificate (Client)* i entrar el nom i l'adreça de correu per la qual volem validar, així com el *passwd* per a protegir-lo fins que arribi al seu nou destinatari (i que després el podrà o l'haurà de canviar). A continuació, hem d'exportar-lo tenint en compte utilitzar el format PKCD#12 que inclou el certificat i la clau. Després d'enviar a l'usuari l'arxiu amb el certificat més el de la *root*-CA, podrà agregar-lo al seu gestor de

correu de manera similar a *root*-CA però com a *personal certificates*. Després, podrà enviar correus signats digitalment i el destinatari (que haurà de tenir instal·lat el certificat de la rootCA) podrà verificar la signatura del correu.

Com a anotació final en relació amb la PKI, tots els nostres usuaris/clients/visitants dels nostres serveis hauran de tenir instal·lat el certificat de la *root*-CA en els seus navegadors/clients per a, d'aquesta manera, validar els llocs/serveis que estan sota el nostre domini/subdominis ja que els navegadors/clients de correu no incorporen aquests certificats arrel per defecte. Si el nostre domini/serveis ho requereix, podem gestionar amb Mozilla la inclusió del nostre certificat en http://www.mozilla.org/en-us/about/governance/policies/security-group/certs/ però no és un tràmit fàcil i és necessari complir amb una sèrie de requisits ja que les garanties del sistema rauen en la inclusió d'aquests certificats.

Activitats

1. Configureu un servidor DNS com a cau i amb un domini propi.

2. Configureu un servidor/client NIS amb dues màquines exportant els directoris d'usuari del servidor per NFS.

3. Configureu un servidor SSH per a accedir des d'una altra màquina sense contrasenya.

4. Configureu un servidor Apache + SSL+ PHP+ MySQL+ PHPAdmin per a visualitzar els fulls personals dels usuaris.

5. Configureu un servidor Apache + un Reverse Proxy per a accedir al servidor web des d'una màquina externa a través del Proxy.

6. Creeu i configureu un sistema de correu electrònic a través d'Exim, Fetchmail, SpamAssassin i un servidor IMAP per a rebre correus des de l'exterior i poder llegir-los des d'una màquina remota amb el client Mozilla (Thunderbird).

7. Instal·leu la wiki MoinMoin i creeu un conjunt de pàgines per verificar el seu funcionament.

8. Instal·leu el servidor de *backups* BackupPC i genereu una còpia de seguretat des d'una màquina Windows i una altra des d'una màquina Linux. Escolliu el mètode de comunicació amb els clients i justifiqueu la resposta.

9. Configureu una CA amb *tiny CA* i genereu/proveu els certificats per a una pàgina web amb SSL i per enviar correus signats i verificar-los des d'un altre compte.

Bibliografia

Tots els URL han estat visitats per última vegada el juny del 2014.

[1] **Debian.org**. *Debian Home*.
<http://www.debian.org>

[2] *Server-World*.
<hhttp://www.server-world.info/en/note?os=Debian_7.0>

[3] **Langfeldt, N.** *DNS HOWTO*.
<http://tldp.org/HOWTO/DNS-HOWTO.html>

[4] **IETF**. Repositori de Request For Comment desenvolupats per Internet Engineering Task Force (IETF) al Network Information Center (NIC). <http://www.ietf.org/rfc.html>

[5] **Instituto de Tecnologías Educativas**. *Redes de área local: Aplicaciones y Servicios Linux*.
<http://www.ite.educacion.es/formacion/materiales/85/cd/linux/indice.htm>

[6] **Trenholme, S.** *MaraDNS - A small open-source DNS server*.
<http://maradns.samiam.org/>

[7] **DNSMasq** *DNS forwarder and DHCP server*.
<https://wiki.debian.org/HowTo/dnsmasq>

[8] *Comparison of FTP client software*
<http://en.wikipedia.org/wiki/Comparison_of_FTP_client_software>

[9] *Servicios de red: Postfix, Apache, NFS, Samba, Squid, LDAP*
<http://debian-handbook.info/browse/es-ES/stable/network-services.html>

[10] *NIS HOWTO*.
<http://www.linux-nis.org/>

[11] **Kukuk, T.** *The Linux NIS(YP)/NYS/NIS+ HOWTO -obs:EOL-*.
<http://tldp.org/HOWTO/NIS-HOWTO/verification.html>

[12] *Exim*.
<http://www.exim.org/docs.html>

[13] *ProcMail*.
<http://www.debian-administration.org/articles/242>

[14] **Apache HTTP** *Server Version 2.2*.
<http://httpd.apache.org/docs/2.2/>

[15] *Apache2 + WebDAV*.
<http://www.debian-administration.org/articles/285>

[16] **Squid** *Proxy Server*.
<http://www.squid-cache.org/ >

[17] **Squid** *Configuration Examples*.
<http://wiki.squid-cache.org/ConfigExamples >

[18] **Kiracofe, D.** *Transparent Proxy with Linux and Squid mini-HOWTO -obs:EOL però interesant en conceptes-*.
<http://tldp.org/HOWTO/TransparentProxy.html#toc1>

[19] **Pinheiro Malère, L. E.** (2007). *Ldap. The Linux Documentation Project*.
<http://tldp.org/HOWTO/LDAP-HOWTO/>

[20] *Proftpd*.
<http://www.debian-administration.org/articles/228>

[21] **PKI** *Public-key cryptography*.
<http://en.wikipedia.org/wiki/Public_key_cryptography >

[22] *SSL/TLS Strong Encryption: An Introduction*.
<http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html#cryptographictech>

[23] *Transparent Multi-hop SSH*.
<http://sshmenu.sourceforge.net/articles/transparent-mulithop.html>

[24] **Christof Paar, Jan Pelzl** *Introduction to Public-Key Cryptography*.
<http://wiki.crypto.rub.de/Buch/movies.php >

[25] **Magnus Runesson** (2007). *Create your own CA with TinyCA2*.
<http://theworldofapenguin.blogspot.com.es/2007/06/create-your-own-ca-with-tinyca2-part-1.html>

[26] textsboldGreg Ferro *Fast Introduction to SOCKS Proxy*.
<http://etherealmind.com/fast-introduction-to-socks-proxy/ >

[27] *Proxy SOCKS*.
<http://en.flossmanuals.net/bypassing-es/proxis-socks/ >

[28] **Mourani, G.** (2001). *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc.
<http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-The-Ultimate-Solution-v2.0.pdf>

[29] *Samba*.
<http://www.samba.org/ >

[30] *Wiki - Samba*.
<https://wiki.samba.org >

[31] *RSAT. Remote Server Administration Tools on a Windows workstation*.
<https://wiki.samba.org/index.php/Installing_RSAT_on_Windows_for_AD_Management>

[32] **M. López, C. Alonso***Samba 4: Controlador Active Directory*.
<http://waytoit.wordpress.com/2013/05/12/samba-4-controlador-active-directory-parte-1-de-3/ >

[33] **M. Rushing***Compiling Samba 4 on Debian Wheezy - Active Directory Domain Controllers*.
<http://mark.orbum.net/2014/02/22/compiling-samba-4-on-debian-wheezy-active-directory-domain-controllers-ho/ >

[34] *Guía Samba4 como Controlador de Dominio y Directorio Activo* .
<http://fraterneo.wordpress.com/2013/08/19/guia-samba4-como-controlador-de-dominio-y-directorio-activo-actualizacion/ >