

Web amb alta disponibilitat i balanceig de càrrega

Francesc Guasch Ortiz

Escola Tècnica Superior d'Enginyers de Telecomunicació de Barcelona
Universitat Politècnica de Catalunya
frankie@etsetb.upc.edu
28 de Maig de 2007

Resum

Instal·lació d'un web capaç de seguir funcionant en cas d'una aturada del servidor. Per aconseguir-ho farem servir una màquina de reemplaç que posarà en funcionament el web si la principal falla. Aquesta segona màquina no estarà aturada, mentre la principal funciona amb normalitat l'aprofitarem per alleugerir la feina del servidor. Les peticions de web es repartiran entre els dos ordinadors. L'usuari té la impressió de què hi ha una sola màquina.

1 Objectiu

L'objectiu és disposar d'un servidor de web que funcioni sempre, o bé la major part del temps que sigui possible. Aquest servei ha de suportar que la màquina on està hostatjat deixi de funcionar. Un sistema dissenyat per garantir un mínim percentatge de temps en funcionament s'anomena d'alta disponibilitat.

Hi han moltes maneres i aplicacions diferents que ens permeten resoldre aquest problema, es proposa una solució que requereix tenir duplicats els diferents components que formen el web, compartir les dades, i posar en marxa automàticament els elements de reserva en cas de fallada.

És poden assolir diferents nivells d'alta disponibilitat, en la nostra proposta ens limitarem a un disseny amb només dos servidors, i ens centrarem en el servei de web amb les noves capacitats incorporades dins les darreres versions del servidor de web Apacheⁱ versió 2.2. Normalment aquest servei requereix també una aplicació de base de dades que també hauria d'estar en un entorn d'alta disponibilitat. En aquest article apuntarem diferents alternatives per la base de dades, però no hi entrarem a fons.

2 Estructura

Per aconseguir l'alta disponibilitat hem de reconèixer que la màquina on tenim instal·lat el web pot fallar i aturar-se. Per superar aquest problema tindrem una altra màquina amb els mateixos serveis instal·lats. Aquest altre ordinador posarà en funcionament el web si el principal falla.

La idea bàsica del sistema és tenir tots els serveis duplicats. Si un dels components falla i no hi ha un altre que agafi el seu lloc provocarà la fallada total del servei. Els components que no estan repetits s'anomenen SPOF (Single Point of Failure).

En primer lloc configurem l'aplicació Apache per fer dues feines diferents: proxy i web. Per tal de donar servei de proxy afegirem uns mòduls especials que ens ho faciliten: mod_proxyⁱⁱ i mod_proxy_balancerⁱⁱⁱ. Aquests mòduls no es fan servir en l'Apache que s'encarrega del web.

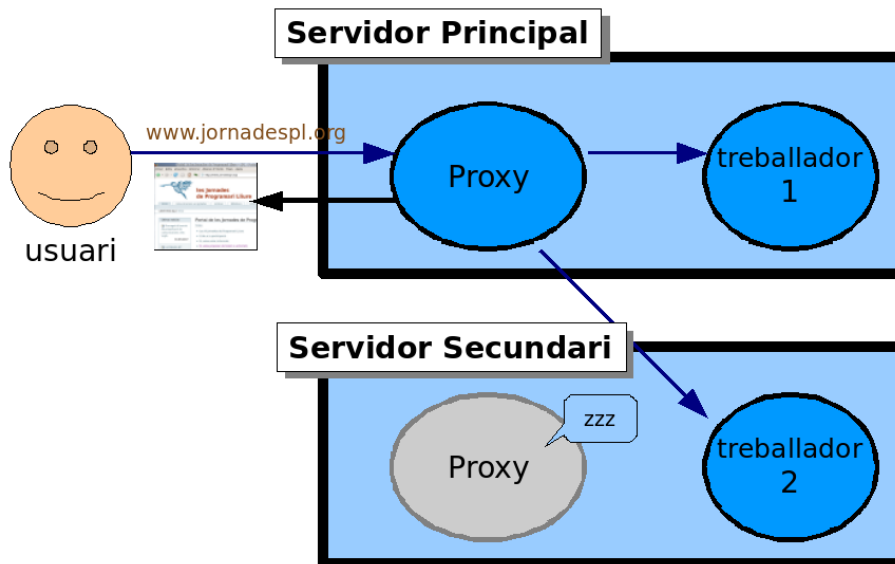


Figura 1: Estructura del Sistema

2.1 Treballador i Proxy

En primer lloc el que abans anomenàvem servidor web a partir d'ara serà el treballador. Segueix encarregant-se de la feina de generar i servir les pàgines web, però ja no parla directament amb els clients.

Afegirem un servei més entre el treballador i el client: un *proxy invers*. L'aplicació proxy s'encarrega d'atendre les peticions, demana el resultat al treballador i els retorna els continguts que han demanat. Pel client res ha canviat, inserir aquesta aplicació no l'afecta en la manera com sol·licita i obté la informació del web. Podríem considerar que el que abans anomenàvem *web*, i que era un servei simple, ara és un conjunt format pel *proxy* i el *treballador*.

Encara que en un principi podria semblar contraproduent afegir aquest pas al mig, té avantatges, principalment:

- Reduïm el temps que el treballador està atenent els clients.
- Permet repartir la càrrega entre diferents treballadors que fan la mateixa feina.
- Protegeix els servidors web de possibles atacs
- Es poden guardar els continguts en memòria cau del proxy

2.1.1 Atenció als clients web

En webs on hi ha moltes peticions simultànies, és convenient que el servidor no sigui el que aten directament als clients. Normalment és un procés molt costós i un dels punts febles és que fins que el client no ha rebut la petició no es pot atendre a una altra. Encara que hi han molts processos fent aquesta atenció, quan abans els alliberem d'aquesta feina millor. Entregant el resultat al proxy es delega en aquesta la tasca d'enviar la informació.

2.1.2 Repartir la feina

Si disposem d'uns quants treballadors web, podem configurar el proxy per que faci les sol·licituds de feina a tots. D'aquesta manera, el client segueix veient el web com un sol punt, però en realitat hi han varies màquines que fan la feina per sota. En les eines que fem servir hi han dues maneres de repartir la feina: comptar peticions i comptar tràfic mesurat.

L'algorisme de comptar peticions funciona enviant les peticions a cada treballador de manera que cada un rep un tant per cent del nombre de peticions. Per exemple és pot posar que el treballador A rebi 70% i el B 30%.

Comptar tràfic mesurat és similar a l'algorisme de comptar peticions però es té en compte la mida del tràfic que es transfereix. Si un treballador està responnent peticions d'arxius molt grans es considera que està fent més feina que l'altre.

2.1.3 Protecció d'atacs

L'accés al web és una finestra oberta a tothom. Qualsevol ordinador del món hi pot accedir. És possible que hi hagi algun punt que sigui vulnerable i que algú que ataquí el nostre servidor web. Afegint una capa proxy entre el client evitem que hi hagi un accés directe cap al servidor. D'aquesta manera alguns dels atacs que es podrien intentar ja no són possibles. Per altra banda, l'accés al servidor encara es pot fer mitjançant el proxy, o sigui que no hem de descuidar la seguretat del web.

2.2 Treballador

El servidor de web segueix funcionant de la mateixa manera que abans d'instal·lar el proxy. Només que ara els clients no hi poden accedir directament. L'administrador sí que ho pot fer per verificar que el servei està funcionant correctament.

2.3 Màquina secundària

Els dos serveis necessaris per fer funcionar el web: proxy i treballador s'instal·len també a una màquina secundària. En aquest servidor tenim en marxa sempre el treballador, i aquest rep les peticions en funció de la repartició que hem configurat al proxy. En situació normal, el proxy està funcionant a la màquina principal i els dos treballadors estan en marxa en ambdues màquines.

En cas de què el proxy de la màquina principal s'aturi es posarà en marxa el de la secundària automàticament.

2.4 Adreces IP

Per poder funcionar aquest sistema, es requereixen tres adreces IP:

1. Adreça IP externa del web.
2. Adreça IP del servidor principal.
3. Adreça IP del servidor secundari.

L'adreça IP del web estarà configurada dins el servidor principal. És on posarem el proxy escoltant les sol·licituds dels clients. Exemple: www.exemple.com

Les altres dues adreces IP són per fer servir internament i no cal ni que siguin públiques. Per exemple, podem fer servir les del rang 192.168.0.x.

El servidor principal tindrà configurades dues adreces IP: la externa i la interna. El servei proxy ha d'atendre les peticions dels clients, o sigui que estarà configurat a l'externa. Els treballadors de web responen directament al proxy, o sigui que els posarem adreces internes.

2.5 Sessions d'usuari

En alguns entorns es requereix que els usuaris mantinguin una sessió. La configuració bàsica del balanceig de càrrega fa que cada petició vagi al treballador que estigui disponible. Per això, la sessió d'un usuari, que consta de diferents peticions, pot ser rebuda per un treballador diferent cada vegada. Aquest sistema funciona perfectament per servidors que guarden el sistema de sessions en una base de dades comú. Quant això no es possible, i volem que els clients siguin atesos sempre pel mateix treballador cal indicar-ho en la configuració del proxy.

2.6 Dades

Com hem vist abans, el proxy reparteix la càrrega fent que les peticions siguin ateses pel treballador que estigui més lliure. Això requereix que tots els servidors de web tinguin la mateixa informació i la mateixa configuració. Per aconseguir-ho farem còpies dels arxius cada cert temps cap al servidor secundari. Amés de la informació en arxius, els webs solen tenir també els continguts en una base de dades. És important tenir en compte que si s'atura la base de dades els usuaris no podran veure la informació correctament. Encara que no és el propòsit d'aquest article farem referència a alguns sistemes d'alta disponibilitat per bases de dades.

2.6.1 Consistència de les Dades

Si dos clients accedeixen simultàniament a dos treballadors distints i sol·liciten les mateixes dades haurien d'obtenir el mateix resultat sempre i quant:

1. La configuració del servidor sigui idèntica a tots els treballadors
2. Els arxius s'hagin replicat amb anterioritat.

La nostra solució depèn d'una aplicació que envia els arxius d'un servidor a l'altre. Per aquesta raó pot donar-se el cas de què canviem un arxiu al servidor principal i no estigui actualitzat al secundari. En cas de servidors en què aquesta consistència sigui vital cal buscar una altre alternativa que ens ho garanteixi.

Per altra banda s'ha de tenir en compte que és molt probable que els continguts dels arxius no siguin els continguts del web. Normalment aquest arxiu són les aplicacions que mostren la informació que hi ha en base de dades. Quant una xifra de la base de dades canvia, tots els treballadors hi podrien accedir simultàniament i obtenen el mateix resultat. Amés l'aplicació de sincronització està implementada d'una manera molt eficient, i pot mantenir sincronitzades una gran quantitat de dades en un mínim temps.

2.6.2 Arxius

Per facilitar aquesta tasca emmagatzemarem totes les dades al servidor principal i seran enviades cap al secundari. Hi han molts sistemes que permeten fer aquest enviament de dades. Una de les més simples i efectives és *rsync*^{iv}. Aquesta aplicació s'executa periòdicament per enviar les dades d'un servidor a l'altre. El principal problema d'aquesta solució és que les dades no es repliquen immediatament al servidor secundari. Hi pot haver un temps de decalatge. Si ho necessitem hauríem de buscar alguna alternativa, per exemple instal·lar SAN^v, NAS^{vi} o iSCSI^{vii}. Comparant amb una replicació feta amb *rsync*, aquestes altres opcions requereixen una alta inversió o compliquen bastant la configuració. En tot cas, s'han de valorar les necessitats i els recursos disponibles a l'hora d'escollir el sistema que més convingui i que permeti a tots els servidors accedir a les mateixes dades.

2.6.3 Base de Dades

És probable que amés dels arxius del web tinguem informació dins una base de dades. Per disposar d'un sistema complert en alta disponibilitat hem de preveure que passaria si s'aturés la base de dades i oferir una solució de recuperació.

Afortunadament hi han bases de dades de codi obert que tenen característiques de replicació, però no garanteixen una recuperació de fallades perfecta. Tant Postgres^{viii} com MySQL^{ix} tenen eines per replicació mestre-esclau, on les dades les envia el mestre cap a l'esclau. Si el mestre falla no hi ha una manera fàcil d'actualitzar els canvis produïts des de l'esclau cap al mestre. Les solucions mestre - mestre són per entorns molt concrets i amb unes característiques limitades. Sembla que MySQL a partir de la 5.1 hauria de proveir replicació avançada, però de moment és una base de dades que encara ha de madurar. També podríem adquirir una base de dades comercial que tingui característiques d'alta disponibilitat, els grans noms de la indústria s'anuncien com capaces de fer-ho, encara que a un cost elevat, i sempre fent servir magatzematge compartit.

Per aquestes raons la millor solució per tenir una base de dades d'alta disponibilitat passa per tenir replicació a nivell de magatzematge. Ja sigui amb una SAN, NAS, iSCSI, o algun sistema de fitxers per xarxa, com DRBD^x.

3 Fallada del servidor

En el nostre exemple tenim dos servidors diferents, el principal i el secundari. Al principal hi ha instal·lat el servei proxy, que atén als clients. El servei web està instal·lat a les dues màquines i està replicat de manera que mostren la informació idèntica.

3.1 Aturada del servidor secundari

En cas de fallada del servidor secundari, el servei donat als usuaris no es veurà afectat. Només ha caigut un dels dos sistemes web. En aquest cas, el proxy veurà que un dels treballadors no funciona i no li demanarà més peticions. Això podria afectar al rendiment en webs amb molt de tràfic. Si realment el nostre web rep moltes consultes i els usuaris patirien en cas d'aturada del servidor secundari, podem considerar instal·lar més de dos treballadors per repartir la càrrega. Els usuaris no s'adonaran de què s'ha aturat un dels treballadors.

3.2 Aturada del servidor principal

La fallada del servidor principal sí que podria provocar problemes, i fer que siguin mínims és l'objectiu de la nostra instal·lació. Si s'atura el servidor principal cal que el secundari prengui el seu lloc. En els dos servidors

tenim instal·lada l'aplicació Heartbeat^{xi} que s'encarrega de verificar que tot està en marxa. En el moment de la fallada del servidor principal, el secundari iniciarà un procés que posa en funcionament el següent:

1. Activarà l'adreça IP del web extern.
2. Es posarà en marxa el servei proxy.

4 Configuracions

Veurem ara exemples de configuracions de les diferents eines que fem servir per garantir l'alta disponibilitat del Web. Després veurem com es fan servir les aplicacions rsync, per duplicar informació entre les màquines, i Heartbeat, per vigilar el bon funcionament dels servidors.

4.1 Proxy

Hi ha diferents aplicacions que ens permeten fer de proxy, en el nostre exemple fem servir Apache, que normalment s'utilitza de servidor web. Fer servir Apache ens permet treballar amb una eina que ja coneixem, i a més aprofitar d'altres característiques disponibles, com traducció d'adreces, balanceig de càrrega, restricció d'accés, o servir documents.

Aquesta és la configuració bàsica del proxy:

4.1.1 Escoltar

Les adreces en les que el proxy està escoltant, en tenim una pel web i una altre pel web segur:

```
Listen www.exemple.com:80
Listen www.exemple.com:443
```

4.2 Registre

Guardem un registre d'accés al servidor per fer estadístiques:

```
CustomLog logs/access_log combined
```

4.2.1 Accés als treballadors

Balanceig de càrrega i reenviament de la feina als treballadors. Tenim un treballador més potent que l'altre, per això el configurem per que accepti més peticions.

```
<VirtualHost _default_:80>
ProxyRequests Off

<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>

ProxyPass / balancer://cluster/
ProxyPassReverse / http://192.168.0.88/
ProxyPassReverse / http://192.168.0.19/

<Proxy balancer://cluster>
    BalancerMember http://192.168.0.88 loadfactor=3
    BalancerMember http://192.168.0.19 loadfactor=7
```

```
ProxySet lbmethod=bytraffic
</Proxy>
</VirtualHost>
```

4.3 Apache

L'únic que es canvia és que ara escoltem només les peticions del proxy, aquest s'encarregarà de parlar amb els clients. En cada un s'haurà de configurar on escolta i forçar la directiva `ServerName`, que normalment és el nom del servidor.

```
Listen 192.168.0.88:80
Listen 192.168.0.88:443
ServerName www.exemple.com
```

4.4 rsync

La transferència de dades es fa mitjançant l'aplicació *rsync*. Per fer-ho executarem la comanda de sincronització cada cert temps mitjançant l'aplicació *cron*. La següent comanda s'afegeix al *cron* de la màquina secundària per què agafi la configuració de la principal. L'usuari administrador del servidor rebrà un missatge cada vegada que hi hagi algun canvi. Posarem una comanda similar per enviar les dades del web.

```
0 * * * mon-fri rsync -avz --delete
principal:/usr/local/web/apache/conf/ /usr/local/web/apache/conf/ |
egrep -v "^(./|$.|sent [0-9]+ bytes|building file list|total size is|
receiving file list)"
```

Podríem dividir la comanda de sincronització en quatre parts: paràmetres, origen, destí i filtratge de la sortida:

- `-avz --delete`: paràmetres que indiquen que ha de comprimir les dades, mostrar informació i esborrar al servidor el que s'hagi esborrat a l'origen.
- `principal:/usr/local/web/apache/conf/` : Màquina i directori origen
- `/usr/local/web/apache/conf/` : Directori destí de la màquina local
- `| egrep -v "^(. :"` : Filtrem el resultat per rebre un missatge cada vegada que s'envia algun canvi. Si hi ha molts enviaments i no volem rebre aquests missatges podem eliminar aquest apartat i afegir el paràmetre `-q` a la comanda *rsync*.

Com estem enviant les dades de configuració del servidor necessitem que s'apliquin al servei els canvis fets. Per això moltes vegades es necessari aturar i tornar a engegar el web. L'aplicació *make* es fa servir normalment per altres coses, però té una característica que ens és molt útil: detecta quant hi ha un canvi en els arxius que li diem i executa comandes en funció d'això. Escrivim un arxiu *Makefile* per dir a *make* que ha d'executar i quins arxius ha de vigilar. Ho farem servir per dues tasques complementàries:

1. Per reiniciar automàticament en cas de canvi de configuració
2. La configuració de l'Apache tindrà una part comú i una part pròpia en cada un dels servidors. Subdividim les directives en dos arxius i els concatenem si canvia alguna.

```
#Makefile
# reinicia el web si canvia la configuració
```

```
logs/httpd.pid: conf/httpd.conf
/etc/init.d/web restart

# concatena els arxius de configuració d'Apache.
conf/httpd.conf: conf/httpd.comu.conf conf/httpd.local.conf
cat conf/httpd.comu.conf conf/httpd.local.conf > conf/httpd.conf
```

Executarem aquesta comanda amb el *cron* per activar el Makefile cada hora.

```
00 08-20 * * * cd /usr/local/web/apache && make -q
```

4.5 Heartbeat

Heartbeat és una aplicació que forma part del projecte Linux-HA^{xii} (Linux d'alta disponibilitat). S'encarrega de detectar servidors aturats, gestiona les comunicacions entre servidors agrupats i posa en marxa i atura els serveis replicats. En el nostre cas vigila si el servei proxy s'ha aturat i l'arrenca a la màquina secundària. És una aplicació bastant complexa, aquí només mostrem un breu resum de la configuració:

- Al directori `/etc/ha.d/resource.d` s'afegeix un enllaç al script que arrenca el proxy:

```
ln -s /etc/init.d/proxy .
```

- A l'arxiu `/etc/ha.rc` es configura quina màquina ha de vigilar. També es descriuen quins servidors estan donant aquest servei redundat, s'anomenen nodes.

```
ucast eth0 principal
node secundaria
node principal
```

5 Millores addicionals

Aquí hem parlat d'un sistema d'alta disponibilitat que requereix un mínim de recursos i hem fet servir les opcions més bàsiques. A banda d'això hi ha moltes coses que es poden fer amb aquesta aplicació:

- Es poden tenir més treballadors per millorar el rendiment del nostre web.
- Podem afegir treballadors que només s'activin si els altres fallen.
- Podríem instal·lar 3 màquines, una fent només de proxy i les altres dues de treballadors, d'aquesta manera, la càrrega del proxy no afectaria al treballador.
- Podem accedir a un gestor del `load_balancer`.
- Podríem fer que el mateix proxy servís continguts, per exemple, un directori amb imatges.
- Podríem tenir un servidor en un altre edifici, encara que si la IP és diferent els servidors de DNS trigarien a actualitzar-se.
- Caldria duplicar també els altres serveis dels que depèn apache, per exemple la base de dades o els dispositius de xarxa.
- Hem fer una configuració de Heartbeat molt simple, es pot millorar afegint una connexió de xarxa dedicada a aquesta tasca entre els dos servidors o afegint un cable serie.
- Es pot configurar el balanceig de manera que les sessions dels usuaris vagin sempre al mateix treballador. Això requereix fer servir *cookies* amb un format especial que el proxy pugui identificar per enviar-lo a cada treballador. Per fer això cal reescriure les peticions fent servir el mòdul d'Apache `mod_rewrite` i afegir aquest identificador^{xiii}.

6 Requeriments i Programari

- Servidor de Web: Apache
- Proxy i balanceig de càrrega: Apache a partir de la versió 2.2
- Replicació dels arxius: rsync, make, cron.
- Gestió dels nodes d'alta disponibilitat: Heartbeat

7 Alternatives

Hem escollit les aplicacions que ens proporcionen l'alta disponibilitat dins un ampli ventall disponible. El nostre procediment ha estat buscar l'eina que ens servia millor i que fos simple i assequible. Descriurem breument alguna d'aquestes altres eines que ens podrien servir per aconseguir uns resultats similars.

7.1 Squid com a proxy

Hem fet servir Apache per dues tasques diferents: proxy i web. Dins la part de proxy el podríem substituir per alguna altra aplicació que té la mateixa funcionalitat de proxy invers. Per exemple, Squid^{xiv} podria configurar-se d'una manera similar. La principal raó d'escollir Apache és que ens és més fàcil mantenir i conèixer a fons una mateixa aplicació, que no aprendre una altre només per fer de proxy.

7.2 Dispositius dedicats com a proxy

Existeixen al mercat diferents empreses que ofereixen productes comercials per fer de proxy i balanceig de càrrega. Aquests sistemes són ordinadors o dispositius de xarxa amb un sistema operatiu tancat. La majoria tenen més característiques que la de fer de simple proxy i recuperació de fallades. Requereixen la instal·lació de treballadors web similar a la que hem creat i moltes vegades la instal·lació de programari en els treballadors per gestionar eficientment el balanceig de càrrega. Cal destacar que aquests productes reparteixen la càrrega d'una manera molt més efectiva que els sistemes de contar que fa servir Apache. El propi treballador és el que indica com està de saturat i la capacitat que té d'acceptar més feina.

7.3 Configuració sense proxy

Una opció més simple que la proposada aquí seria no fer servir cap aplicació proxy i duplicar el servidor web en la màquina secundària. Aquest web estaria aturat a menys que el principal caigués. Hauríem configurat Heartbeat per activar-lo si el principal s'atura d'una manera similar a com ho hem fet amb el proxy. No s'ha escollit aquesta opció per que desaprofitaríem els recursos del servidor secundari que estaria en marxa sense fer res, només esperant si falla el principal.

7.4 Balanceig de càrrega amb *pen*

Pen^{xv} és una aplicació per balanceig de càrrega que serveix per algunes aplicacions que donen serveis mitjançant el protocol de xarxa TCP, com el web. És de configuració molt simple i una característica interessant és que sempre envia els clients als mateixos treballadors, si es que estan disponibles.

7.5 Linux Virtual Server

El nostre objectiu era poder recuperar la fallada del servidor web, el projecte Linux Virtual Server^{xvi} (LVS) té una orientació més ambiciosa i pretén proporcionar alta disponibilitat de tot el servidor linux.

8 Conclusió

Hem instal·lat un sistema web d'alta disponibilitat amb Apache-2.2 amb una mínima inversió i fent servir eines de programari lliure com Heartbeat, cron, rsync i make. Tenir aquest entorn no serveix només en cas d'una fallada del servidor. També facilita la feina de l'administrador. Podem aturar un servidor, ajustar paràmetres de configuració, provar-lo fora del web per que no hi accedeixin els clients, i un cop hem comprovat que tot funciona correctament, tornar-lo a afegir.

S'ha proposat sincronitzar els arxius i les configuracions des del servidor principal cap al secundari. Aquesta solució fa que hi hagi un temps de retràs fins que totes les dades s'han replicat. En la pràctica dona uns resultats excel·lents amb un mínim esforç, gràcies als algorismes interns de replicació molt eficients. Cal una acurada planificació de les replicacions i ajustar les possibilitats que ens donen les eines a les nostres necessitats.

- i Apache: <http://httpd.apache.org>
- ii mod_proxy: http://httpd.apache.org/docs/2.2/mod/mod_proxy.html
- iii mod_proxy_balancer: http://httpd.apache.org/docs/2.2/mod/mod_proxy_balancer.html
- iv rsync: <http://samba.anu.edu.au/rsync/>
- v SAN: http://en.wikipedia.org/wiki/Storage_area_network
- vi NAS: http://en.wikipedia.org/wiki/Network-attached_storage
- vii iSCSI: <http://en.wikipedia.org/wiki/Iscsi>
- viii Postgres: <http://www.postgresql.org/>
- ix MySQL: <http://www.mysql.com/>
- x DRBD: <http://www.drbd.org/>
- xi Heartbeat: <http://www.linux-ha.org/Heartbeat>
- xii Linux-HA: <http://www.linux-ha.org/>
- xiii Balanceig de càrrega i sessions http://howtoforge.com/load_balancing_apache_mod_proxy_balancer
- xiv Squid: <http://www.squid-cache.org/>
- xv Pen: <http://siag.nu/pen/>
- xvi Linux Virtual Server: <http://www.linuxvirtualserver.org/>