Instalimi dhe Konfigurimi i nje Serveri Gateway te Thjeshte

Dashamir Hoxha

gateway-server-en-0.1 (2004-10-01)

Table of Contents

1. Hyrje

1.1. About this tutorial

This tutorial is based on my experience with installing and configuring a small gateway server. It is going to be useful to me in case that I will do a similar installation in the future, and it may also be useful to other people that have to do something like this. It takes a practical approach, describing all the steps in details, with all the commands that are used in the concrete example.

1.2. Who should read this tutorial

Anybody that wants to learn how to install and maintain a linux server can read this tutorial. However, it assumes that the reader has some experience with linux and it does not explain everything in details. So, if you have no previous experience with linux, it will be hard to understand and to follow the instructions in the guide, and you have better to start with some introductory tutorials first.

In order to help you to understand whether you can follow this guide easily, try to answer the following questions. If you can answer them positively, then most probably you can follow this guide easily. Otherwise, maybe you have better to study some other guides or tutorials before this one.

1. Have you ever used linux? Do you know what is Fedora?
2. Have you ever installed a linux system yourself? Do you know what is a partition? Do you know what is a *swap* partition?
3. Have you ever used the commands of linux? Do you know what is a *terminal* ? Have you ever used *ls* , *cd* , *mkdir* , *cp* , *rm* ?
4. Do you know what is *bash* ? Do you know what are *shell scripts* ?

5. Have you ever configured a network interface yourself? Do you know what is an *IP* and a *netmask* ?

6. Do you know what are services? Have you ever heard about *DNS* , *sendmail* , *apache* ?

7. Have you ever used the *vi* editor?

1.3. Description of the situation

This document describes the installation and configuration steps for the server that is installed at the network of a small company or institution. It serves as the gateway of the network (protecting it with a firewall), as the web server of the client, as a port forwarder to the CACHE(DB) server, etc.

The diagram below shows how the components of the network are connected physically. There is a router-modem that provides connection to the Internet. The router and the computers are connected in an ethernet network by means of a HUB (which is represented in the picture by the thick line).

Physical diagram of the network

What we want to do is to place a linux server between the router and the rest of the local network, so that it can protect it with a firewall. The linux server will also serve as the web server of the company/organization.

We will do this in two steps. First we are going to configure the server according to the following diagram:

First configuration of the network

In this network configuration the linux server can perform all of its functions: gateway, web server, port-forwarding, etc. However the other computers can choose either the linux server, or the router itself as gateway. This is done in order not to interrupt the Internet connection for the local network during the time that the server is installed and tested. Also, the switch to the new gateway is done gracefully, without Internet connection interruption.

Once the server is installed and tested, and once all the computers switch to the new gateway, we can change the configuration of the network as shown in the diagram below:

Second configuration of the network

In this configuration the local network can access the Internet only through the linux server. During the configuration of the server we will take care so that we can switch instantly from the first configuration to the second configuration (we will see how). Also, the local computers don't need to change the gateway, it is the same as before: 10.10.3.100. The physical connection doesn't need to be changed as well; the router, the server and the local machines are still connected to the HUB, same as before.

1.4. Requirements

The minimal requirements for the linux server are:

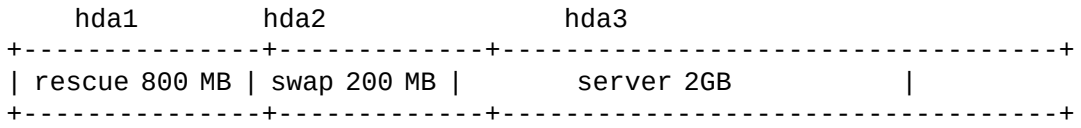
- Pentium I, CPU
- 64MB RAM
- 2GB HDD
- Floppy drive (no need for CD-ROM drive)
- Two ethernet network cards.

If you have hardware older than this, you can still try it, it may work.

2. Installation

The harddisk of the server is 3GB and it is partitioned as seen in the figure.

Figure 1. Partitions of the disk



The first partition holds a minimal linux installation, which can be used as a rescue system in case that something goes wrong in the server and it becomes inaccessible. The swap partition is about twice as the size of RAM. The server partition must be about 2GB. The rescue partition can be omitted, however the configurations described here assume the above configuration of the disk partitions.

The installation is based on Fedora1 and it is done from Internet, using a boot floppy and a network drivers floppy.

2.1. Prepare the Installation Server

An installation server contains the installation disks of linux and makes them accessible through NFS or FTP or HTTP. Our installation server will make available Fedora1 through HTTP.

1. First download linux. I will describe a scenario below however it may change according to the distribution, ftp server, etc. For a list of download mirrors where you can find Fedora, see http://fedora.redhat.com/download/mirrors.html .

```
bash# mkdir /var/local/fedora1
bash# cd /var/local/fedora1
bash# lftp ftp://ftp.easynet.nl
lftp ftp.easynet.nl:~> cd mirror/fedora/1/i386/iso/
lftp ftp.easynet.nl:~> ls
-rw-r--r--  1 ftp    easynet 669122560 Sep  3 21:54 yarrow-i386-disc1.iso
-rw-r--r--  1 ftp    easynet 677511168 Mar 14  2003 yarrow-i386-disc2.iso
-rw-r--r--  1 ftp    easynet 508592128 Mar 14  2003 yarrow-i386-disc3.iso
lftp ftp.easynet.nl:~> at 00:30 tomorrow -- get yarrow-i386-disc1.iso &amp;
lftp ftp.easynet.nl:~> at 02:30 tomorrow -- get yarrow-i386-disc2.iso &amp;
lftp ftp.easynet.nl:~> at 04:30 tomorrow -- get yarrow-i386-disc3.iso &amp;
lftp ftp.easynet.nl:~> exit bg
```

The download is scheduled for after midnight, so that it does not block the network traffic.

2. Create a script that will mount the Fedora1 ISO disks:

```
bash# vi /var/local/fedora1/mount-fedora-discs.sh
```

```
#!/bin/bash

cd /var/local/fedora1
mount -t iso9660 -o loop yarrow-i386-disc1.iso disc1/
mount -t iso9660 -o loop yarrow-i386-disc2.iso disc2/
mount -t iso9660 -o loop yarrow-i386-disc3.iso disc3/
```

Make it executable:

```
bash# chmod 755 /var/local/fedora1/mount-fedora-discs.sh
```

3. Mount the Fedora1 ISO disks:

```
bash# cd /var/local/fedora1/
bash# mkdir disc1
bash# mkdir disc2
bash# mkdir disc3
bash# ./mount-fedora-discs.sh
```

Mount them also each time that the computer is rebooted:

```
bash# vi /etc/rc.d/rc.local

### mount fedora1 discs
/var/local/fedora1/mount-fedora-discs.sh
```

4. Make them accessible from the web:

```
bash# cd /var/www/html/
bash# mkdir fedora1
bash# cd fedora1/
bash# lndir /var/local/fedora1/disc1
bash# lndir /var/local/fedora1/disc2
bash# lndir /var/local/fedora1/disc3
```

Note

Make sure that *apache* is up and running:

```
bash# /sbin/chkconfig --list httpd
bash# /sbin/chkconfig --level 2345 httpd on
```

```
bash# /sbin/service httpd status
bash# /sbin/service httpd restart
```

2.2. Prepare Installation Floppies

To prepare the floppies we will use the floppy images that we can download from the installation server. The IP *11.22.33.44* that is used as the IP of the installation server, should be replaced by the IP of your installation server. In case that you couldn't prepare an installation server (for any reason), you can use any public HTTP installation server that can be found at http://fedora.redhat.com/download/mirrors.html . For example, instead of `http://11.22.33.44/fedora1/` you can use http://mirrors.kernel.org/fedora/core/1/i386/os/ .

To prepare the floppies, follow these steps:

1. Download `bootdisk.img` and `drvnet.img` from http://11.22.33.44/fedora1/images/
2. In linux, use the command **dd** to write the floppies, like this:

```
bash$ dd if=bootdisk.img of=/dev/fd0 bs=1440k
bash$ dd if=drvnet.img of=/dev/fd0 bs=1440k
```

3. In windows, download `rawrite.exe` from http://11.22.33.44/fedora1/dosutils/ , and use it like this:

```
C:> rawrite
Enter disk image source file name: bootdisk.img
Enter target diskette drive: a:
Please insert a formatted diskette into drive A: and
press --ENTER-- : [Enter]
C:>
```

2.3. Install Rescue

The installation of *Rescue* can be done using the first installation floppy. At the `boot:` prompt type **linux dd** :

```
boot: linux dd
```

The parameter *dd* tells to the kernel that we have a *driver disk* , so, after the kernel is loaded, it will ask for driver floppies. At this time we insert the network drivers floppy (the second floppy), and after that the kernel will be able to configure and use the network cards, so that it can perform the rest of the installation from the Internet.

Before continuing with the installation, it will ask for the installation method (select *HTTP*), for the installation server and path (set http://11.22.33.44/fedora1/), for the parameters of the network card and the network (IP, NETMASK, GATEWAY, DNS SERVER, etc.). It will retrieve the installation program from the installation server and the rest of the installation is just like a normal (CD-ROM) installation. Make sure that when you select the packages, you select no packages at all; this does not mean that no packages at all will be installed, it means that only the most necessary packages will be installed and no additional packages (a minimal installation).

Alternatively, the installation of the *Rescue* system can be done almost automatically like this:

1. Create a *kickstart file* :

```
bash$ vi ks.cfg

# Kickstart file
network --device eth0 --bootproto static --ip 10.10.3.101
      --netmask 255.255.255.0 --gateway 10.10.3.253
      --nameserver 11.22.33.44 --hostname rescue
network --device eth1 --bootproto static --ip 10.10.3.100
      --netmask 255.255.255.0 --gateway 10.10.3.253
      --nameserver 11.22.33.44 --hostname rescue
url --url http://11.22.33.44/fedora1
lang en_US.UTF-8
langsupport --default en_US.UTF-8 en_US.UTF-8
timezone Europe/Tirane
keyboard us
install
rootpw --iscrypted $1$FpTm50gJ$gL82MlzncIjZA6MPUGkBD/
firewall --enabled --http --ssh
authconfig --enableshadow --enablemd5
mouse none
skipx
bootloader --location=mbr --lba32
#use existing partitions
part / --fstype ext3 --onpart hda3
part swap --onpart hda2
reboot

%packages

%post
```

2. Write it to the first floppy:

```
bash$ mount /mnt/floppy
bash$ cp ks.cfg /mnt/floppy
```

3. At the time of installation write this at the `boot:` prompt:

```
boot: linux dd ks=floppy
```

Then the rest of the installation should continue without intervention.

Note

It is better to write `ks.cfg` in both floppies, in case that you forget to swap the floppies after the network drivers are loaded.

2.4. Install Server

1. Create the file `ks-server.cfg` and write it on both floppies. If there is not enough space in the first floppy, remove some of the files `*.msg` .

2. Modify the file `syslinux.cfg` in the boot floppy (first floppy) by adding these lines:

```
default server
prompt 1
timeout 300
[. . . . .]
label server
  kernel vmlinuz
  append dd ks=floppy:/ks-server.cfg initrd=initrd.img ramdisk_size=8192
[. . . . .]
```

   (It may be readonly, make it writable first.)

3. Reboot the machine with the boot floppy in and write at the prompt:

```
boot: server
```

   or just pres [enter], since it is the default.

4. Wait until the installation is finished.

2.5. ks-server.cfg

The post-install script at the end of the file does the configuration of the server after installation. However, the configuration steps will be described in detail later, so that they can be done manually, in case that something goes wrong at the installation time.

```
# Kickstart file
network --device eth0 --bootproto static --ip 10.10.3.101 --netmask
255.255.255.0 --gateway 10.10.3.253 --nameserver 11.22.33.44 --hostname gateway
network --device eth1 --bootproto static --ip 10.10.3.100 --netmask
255.255.255.0 --gateway 10.10.3.253 --nameserver 11.22.33.44 --hostname gateway
url --url http://11.22.33.44/fedora1
lang en_US.UTF-8
langsupport --default en_US.UTF-8 en_US.UTF-8
timezone Europe/Tirane
keyboard us
install
rootpw --iscrypted $1$FpTm50gJ$gL82MlzncIjZA6MPUGkBD/
firewall --enabled --http --ssh
authconfig --enableshadow --enablemd5
mouse none
skipx
bootloader --location=mbr --lba32
#use existing partitions
part / --fstype ext3 --onpart hda3
part swap --onpart hda2
reboot
```

```
%packages --resolvedeps
@ web-server
@ mail-server
@ dns-server
@ dialup
@ sql-server
@ editors
@ admin-tools
@ system-tools
@ ftp-server
@ text-internet
grub
kernel
samba
samba-swat

%post

### This script configures the system after installation

### downloaded them from the web
cd /
wget http://11.22.33.44/backup/config-webserver.tgz
wget http://11.22.33.44/backup/cache-5.0.4-1.i386.rpm

### alternatively, if they are backed up in the Rescue partition,
### mount /dev/hda1 in order to access the config backup and the cache rpm
# mkdir /mnt/hda1
# mount /dev/hda1 /mnt/hda1


### restore some configuration files
cd /
tar xvpfz config-webserver.tgz
# tar xvpfz /mnt/hda1/backup/config-webserver.tgz

### install boot menu
grub-install --force-lba /dev/hda

### start httpd
/sbin/chkconfig --level 2345 httpd on
/sbin/service httpd start

### configure and start samba
/usr/sbin/useradd samba
chgrp samba /var/www/html
chmod 775 /var/www/html
/sbin/chkconfig --level 2345 smb on
/sbin/service smb start

### install cache
rpm -Uhv cache-5.0.4-1.i386.rpm
# rpm -Uhv /mnt/hda1/backup/cache-5.0.4-1.i386.rpm
```

```
# the rest of cache installation must be done manually
# bash# ssh root@localhost
# bash# cd /usr/cachekit/5.0.4/
# bash# ./cinstall
```

2.6. Fix GRUB (bootloader menu)

The bootloader menu will be installed in the Rescue partition, because it is safer there. Follow these steps:

1. Open the file `/boot/grub/grub.conf` and edit it so that it looks like this:

```
bash# vi /boot/grub/grub.conf

default=1
timeout=10
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
title Fedora Core (2.4.22-1.2115.nptl)  rescue
      root (hd0,0)
      kernel /boot/vmlinuz-2.4.22-1.2115.nptl ro root=/dev/hda1
      initrd /boot/initrd-2.4.22-1.2115.nptl.img
title Fedora Core (2.4.22-1.2115.nptl)  gateway
      root (hd0,2)
      kernel /boot/vmlinuz-2.4.22-1.2115.nptl ro root=/dev/hda3
      initrd /boot/initrd-2.4.22-1.2115.nptl.img
```

2. Now **mount** the rescue partition and **chroot** there:

```
bash# mkdir /mnt/hda1
bash# mount /dev/hda1 /mnt/hda1
bash# /usr/sbin/chroot /mnt/hda1
```

3. After **chroot** , we are in the Rescue system. Modify the file `/boot/grub/grub.conf` same as above.

4. Now re-install grub (it was installed first during installation):

```
bash# /sbin/grub-install --help
bash# /sbin/grub-install --force-lba /dev/hda
```

3. Network Configuration

The network configuration is done by the script `network-setup` and by the configuration file `network-config` , which is included and used by this script (and other configuration scripts as well). The server has two network interfaces, one external (eth0) and one internal (eth1). For the network configuration of the server we need to know the IP and NETMASK of these interfaces, the GATEWAY to internet, and the IP of the CACHE (database) server to which we are going to do port forwarding. This information is stored in the file `network-config` .

This way of separating the configuration information from the configuration scripts gives us flexibility for changing the configuration quickly and safely. For example, if we need to change the gateway, all we should do is to modify `network-config` accordingly and to run the script `network-setup` :

```
bash# vi /usr/local/config/network-config
bash# /usr/local/config/network-setup
```

However, if we have some standard configurations that are repeated time after time, then it is better to prepare some standard config files and use one of them accordingly, like this:

```
bash# cd /usr/local/config/
bash# cp network-config-1 network-config
bash# ./network-setup
```

3.1. /usr/local/config/network-setup

```
#!/bin/bash

### include the configuration file
. /usr/local/config/network-config

### set up the links, in  case that they are not up
/sbin/ip link set eth0 up
/sbin/ip link set eth1 up

### flush any existing ip addresses of eth0 and eth1
/sbin/ip address flush eth0
/sbin/ip address flush eth1

### add new addresses
/sbin/ip address add $ETH0_IP/$ETH0_MASK dev eth0
/sbin/ip address add $ETH1_IP/$ETH1_MASK dev eth1

### add a default route
/sbin/ip route add default via $GATEWAY dev eth0
```

3.2. /usr/local/config/nework-config

```
### internal interface of the router
GATEWAY=192.168.0.1

### external interface of the web server
ETH0_IP=192.168.0.201
ETH0_MASK=24

### internal interface of the web server
ETH1_IP=192.168.0.200
ETH1_MASK=24

### database server (cache) interface
CACHE_IP=192.168.0.10
```

3.3. /usr/local/config/network-config-1

```
### internal interface of the router
GATEWAY=10.10.3.253

### external interface of the web server
ETH0_IP=10.10.3.101
ETH0_MASK=24

### internal interface of the web server
ETH1_IP=10.10.3.100
ETH1_MASK=24

### database server (cache) interface
CACHE_IP=10.10.3.102
```

3.4. /usr/local/config/network-config-2

```
### internal interface of the router
GATEWAY=192.168.0.1

### external interface of the web server
ETH0_IP=192.168.0.2
ETH0_MASK=30

### internal interface of the web server
ETH1_IP=10.10.3.100
ETH1_MASK=24
```

```
### database server (cache) interface
CACHE_IP=10.10.3.102
```

3.5. /usr/local/config/network-config-3

This is a configuration used in another network. Suppose that we install and test the server in one network and deploy it in another one. The flexibility of the configuration scripts allows us to do this.

```
### internal interface of the router
GATEWAY=192.168.0.1

### external interface of the web server
ETH0_IP=192.168.0.201
ETH0_MASK=24

### internal interface of the web server
ETH1_IP=192.168.0.200
ETH1_MASK=24

### database server (cache) interface
CACHE_IP=192.168.0.10
```

4. Firewall

In the firewall we will accept only the HTTP port (80), ssh port (22), FTP ports (20, 21), samba ports (137,138,139,445) and the port that will be forwarded to the CACHE server (1972). Everything else will be rejected. Edit the file `/etc/sysconfig/iptables` (don't be affraid about the manual customization not being recommended, it is not forbidden, and you can customize it if you know what you are doing):

```
bash# vi /etc/sysconfig/iptables

# Firewall configuration written by redhat-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
```

```
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

### accept HTTP port: 80
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT

### accept FTP ports: 20 and 21
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 20 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT

### accept ssh port: 22
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT

### accept cache port: 1972
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 1972 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 1972 -j ACCEPT

### accept samba ports
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT

-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

Now restart the iptables to apply these rules:

```
bash# /sbin/service iptables restart
bash# /sbin/iptables-save | more
```

5. Source NAT (Masquerading)

Source Network Address Translation (SNAT) is a trick (technique) that makes the server act as a proxy for all the networks that uses it as a gateway. This is useful for protecting the local network from the outside attacks, if you own only one real IP, etc.

The script that is used to start or stop SNAT-ing is `/usr/local/config/source-nat` :

```
bash# vi /usr/local/config/source-nat

#!/bin/bash
# enable or disable source NAT (masquerading)

### include the network configuration
. /usr/local/config/network-config

case "${1}" in
```

```
  ls )
    /sbin/iptables-save --table nat
    exit 0
    ;;

  flush )
    /sbin/iptables --verbose --table nat --flush
    /sbin/iptables --verbose --table nat --delete-chain
    exit 0
    ;;

  on )
    echo 1 >/proc/sys/net/ipv4/ip_forward
    /sbin/iptables --verbose --table nat --append POSTROUTING \
                --out-interface eth0 \
                --jump SNAT --to-source $ETH0_IP
          # --jump MASQUERADE
    exit 0
    ;;

  off )
    echo 0 >/proc/sys/net/ipv4/ip_forward
    /sbin/iptables --verbose --table nat --delete POSTROUTING \
                --out-interface eth0 \
                --jump SNAT --to-source $ETH0_IP
                # --jump MASQUERADE
    exit 0
    ;;

  * )
    echo "Usage: ${0} [ ls | flush | on | off ]"
    exit 0
    ;;
esac
```

Make it executable and then start SNAT-ing like this:

```
bash# cd /usr/local/config
bash# chmod 755 source-nat
bash# ./source-nat
bash# ./source-nat on
```

6. Web Server

```
bash# /sbin/chkconfig --level 2345 httpd on
bash# /sbin/service httpd start
```

7. CACHE

1. Go to http://www.intersystems.com/cache/downloads/ and download a FREE, fully functional, non-expiring, single-user Cache. Then install the cache rpm package:

```
bash# wget http://11.22.33.44/backup/cache-5.0.4-1.i386.rpm
bash# rpm -Uhv cache-5.0.4-1.i386.rpm
```

2. Login as *root* , otherwise the cache installation program will refuse to run:

```
bash# ssh root@localhost
```

3. Install CACHE:

```
bash# cd /usr/cachekit/5.0.4/
bash# ./cinstall

Enter configuration name: CACHE
Do you want to create configuration 'CACHE' &lt;No&gt;? yes
Enter destination directory name for this installation.
Directory: /usr/cachesys
Directory '/usr/cachesys' does not exist.
Do you want to create it &lt;No&gt;? yes

Disk blocks required  = 373056
Disk blocks available = 513312

Do you want to install Cache Unicode support &lt;No&gt;? yes

Do you want to configure your Web Server for CSP &lt;Yes&gt;?

Do you want to install ODBC and SQL Gateway &lt;Yes&gt;?

Do you want to load the Manager utility source code &lt;No&gt;?

Do you want to load the Cache Engine link libraries &lt;No&gt;?

What group should be allowed to start and stop Cache
root

[. . . .]
```

4. Configure CSP (CACHE Server Pages) by modifying `/usr/cachesys/csp/CSP.ini` appropriately.

5. Start CSP:

```
bash# cd /usr/cachesys/csp/
bash# ./CSPnsd
```

6. Add the commands above (for starting CSP) in `/etc/rc.d/rc.local` so that they are executed whenever the server is rebooted.

7.1. /usr/cachesys/csp/CSP.ini

```
[SYSTEM]
Server_Response_Timeout=60
Queued_Request_Timeout=60
Run_Time_Script=CSPcgi
Default_Server=cachesrv

[SYSTEM_INDEX]
LOCAL=Enabled
cachesrv=Enabled
SYSTEM=Enabled

[LOCAL]
Ip_Address=127.0.0.1
TCP_Port=1972
Minimum_Server_Connections=3

[APP_PATH_INDEX]
/=Enabled
/csp=Enabled

[APP_PATH:/]
Default_Server=cachesrv
Alternative_Servers=Disabled

[APP_PATH:/csp]
Default_Server=cachesrv
NameSpace=Samples
Alternative_Servers=Disabled

[cachesrv]
Ip_Address=192.168.0.10
TCP_Port=1972
Minimum_Server_Connections=3
Maximum_Server_Connections=65000
Server_Response_Timeout=60
```

8. CACHE Port Forwarding

Computers in the local network cannot be accessed from outside, because the proxy server is the only visible point of the local network. However, we need to access from outside the CACHE server (port 1972) which is in a machine behind the proxy server. This is done using port forwarding; whenever the proxy server gets a packet with a destination port 1972, it forwards it to the CACHE server in the local network (this is called DNAT -- Destination Network Address Translation). Then, the packets that come from the CACHE server are forwarded to the outside machine. The outside machine thinks that it is the proxy that is replying to it, although in fact it is a machine behind the proxy that handles its request.

Port forwarding is enabled by the script `/usr/local/config/port-forward` , which is as follows:

```
bash# vi /usr/local/config/port-forward

#!/bin/bash
# forward the port to the DB server
PORT=1972

### include the network configuration file
. /usr/local/config/network-config

### enable forwarding
echo 1 >/proc/sys/net/ipv4/ip_forward

### forward the port
/sbin/iptables -t nat -A PREROUTING \
           -p tcp -d $ETH0_IP --dport $PORT \
           -j DNAT --to-destination $CACHE_IP

### in case that $ETH1 is not gateway for $CACHE_IP
### but it doesn't hurt even if $ETH1 is gateway for it
/sbin/iptables -t nat -A POSTROUTING \
           -p tcp -d $CACHE_IP --dport $PORT \
           -j SNAT --to-source $ETH1_IP
```

9. Samba

Samba is used to share the DocumentRoot of the web server, so that everybody in the local network can update the content of the web pages from their windows machines.

1. Modify the following variables in `/etc/samba/smb.conf` , in the group *[global]* , like this:

```
[global]
  workgroup = WERELD
  server string = Web Server
  hosts allow = 10.10.3. 192.168. 127.
  guest account = samba
  security = share
```

2. Add the group *[public]* at the end of it:

```
[public]
  path = /var/www/html/
  public = yes
  only guest = yes
  writable = yes
  printable = no
```

3. Create the user *samba* and set permissions to `/var/www/html` :

```
bash# /usr/sbin/useradd samba
bash# chgrp samba /var/www/html
bash# chmod 775 /var/www/html
```

4. Start the samba service:

```
bash# /sbin/chkconfig --level 2345 smb on
bash# /sbin/service smb start
```

5. Don’t forget that the samba ports should be accepted in firewall:

```
bash# vi /etc/sysconfig/iptables

### accept samba ports
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 137 -j
ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 138 -j
ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j
ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j
ACCEPT
```

10. Reconfiguration

In order to make the configurations permanent, add these lines to `/etc/rc.d/rc.local`, so that they are executed each time that the server is rebooted:

```
bash# vi /etc/rc.d/rc.local

### configure network interfaces
/usr/local/config/network-setup

### CACHE port forwarding
/usr/local/config/port-forward

### source NAT (masquerading)
/usr/local/config/source-nat on

### start the CSP service
cd /cachesys/csp/
./CSPnsd
```

For updating the configuration of the server without rebooting (e.g. when something is modified in the configuration files), the script `/usr/local/config/reconfig` is used:

```
bash# vi /usr/local/config/reconfig

#!/bin/sh
### reconfig the network after changing the network-setup

### configure network interfaces
/usr/local/config/network-setup

### restarting iptables will flush all the rules
### (also the rules about forwarding and SNAT)
/sbin/service iptables restart

### CACHE port forwarding
/usr/local/config/port-forward

### source NAT (masquerading)
/usr/local/config/source-nat on

### restart cache
cd /cachesys/csp/
./CSPnsd stop
./CSPnsd
```

1. Copy `/usr/local/config/network-config-1` or `/usr/local/config/network-config-2` to `/usr/local/config/network-config` , in order to enable this configuration and disable the other.

2. In case that the port forwarded to CACHE needs to be changed, change it at the `/usr/local/config/port-forward` script and to `/etc/sysconfig/iptables` .

3. Also, if the CACHE_IP changes, change it at `/cachesys/csp/CSP.ini` as well.

4. Run the script `/usr/local/config/reconfig` in order to change the network configuration immediately, or reboot.

11. Backup Configuration Files

Most of the configuration files and scripts that are specific for this webserver, are kept in the directory `/usr/local/config/` , so that they can be accessed and backed up easily. Even the config files that are not in this directory have a symbolic link inside it for easy access and for having a clear idea of the config files that are modified after the installation. A listing of this directory looks like this:

```
bash$ ls /usr/local/config/
backup.sh       network-config   network-setup   reconfig
config-files.txt  network-config-1  port-forward    smb.conf
CSP.ini        network-config-2  post-install.sh  source-nat
iptables       network-config-3  README
```

The backup of configuration files and scripts is done by the script `/usr/local/config/backup.sh` :

```
#!/bin/bash

tar cvpfz config.tgz --files-from=/usr/local/config/config-files.txt
```

The script backs up all the files that are listed in the file `/usr/local/config/config-files.txt` :

```
bash# vi /usr/local/config/config-files.txt

/usr/local/config/
/etc/sysconfig/iptables
/etc/rc.d/rc.local
/var/cache/csp/CSP.ini
/etc/samba/smb.conf
/boot/grub/grub.conf
```

In order to perform another installation of a server like this (or a re-installation of the same server), the backup file `config.tgz` (and cache rpm as well) is placed in the installation server:

```
bash$ ssh dasho@11.22.33.44 mkdir -p /var/www/html/backup/
```

```
bash$ scp backup.tgz dasho@11.22.33.44:/var/www/html/backup/
bash$ scp cache-5.0.4-1.i386.rpm dasho@11.22.33.44:/var/www/html/backup/
```

Also, they can be backed up in the Rescue partition (together with other useful data), so that they can be accessed easily in case of re-installation:

```
bash# mkdir /mnt/hda1
bash# mount /dev/hda1 /mnt/hda1
bash# mkdir /mnt/hda1/backup
bash# cp backup.tgz /mnt/hda1/backup/
bash# cp cache-5.0.4-1.i386.rpm /mnt/hda1/backup
bash# umount /mnt/hda1
```