Procedimiento para instalar la sala de sistemas del SSF+UN

William Fdo. Oquendo Patiño*

March 26, 2014

Contents

1 Generalidades

- LA ÚNICA MANERA DE ENTENDER LO QUE SE MUESTRA ACÁ Y DE MEJORARLO ES LEYENDO LA DOCUMENTACIÓN. POR FAVOR LEA LOS MANUALES, LOS HOWTO Y APÓYESE EN GOOGLE.
- La distribución a instalar es Slackware, Versión 13.1 a la fecha.
- Sistemas de 32 bits. Sistemas de 64 bits quedarán con OS de 32. Es posible mezclar 64 y 32, pero eso no se tratará en este documento.
- Los servicios a configurar son: red interna con DNS MASQ (DHCP + DNS + NTP + MASQUERADING) , NFS, NIS, FIREWALL,, BACKUP.
- Se supone que el servidor tiene dos tarjetas de red: una para la red interna y otra para la externa. La tarjeta de red que conecta con la red exterior se denotará por ETH0, la tarjeta de red de la red interna se denotará por ETH1. Tambien se supone la existencia de unn hub al que se conecta la red interna. La externa se conecta al punto de red.
- El segmento de red utilizado en este ejemplo será 192.168.123.xxx , y se supone que el dominio escogido será ssf.net
- Al final del documento se muestran algunos ejemplos de los archivos de configuración.

2 Instalación de Slackware

Es necesario tener configurado cada computador con las particiones apropiadas. Si es sólo linux, se sugieren mínimo tres particiones: Una para el sistema, otra para la swap, otra para el home. Si se busca tener linux y windows se sugieren mínimo 5: las tres antes mencionadas y una para Windows y talvez otra para intercambio, aunque en los linux modernos el driver ntfs-3g permite lectura escritura en ntfs y por tanto la particiónd e intercambio no es necesaria.

Al final de esta sección se supone que el ususario ha instalado exitosamente Slackware en cada computador, incluido el servidor. El sistema se ha configurado para obtener ip por DHCP.

La instalación de Slackware está bien descrita en varias páginas web, se remite al lector a google para encontrar la información pertinente.

3 Instalación por red usando bootp, tftp y pxe

source: usb-pxe slackware installation howto, http://alien.slackbook.org/dokuwiki/doku.php?id=slackware:pxe

*woquendo@gmail.com

3.1 Server PXE, TFTP, etc

We are going to use dnsmasq. The mirror of slackware will be on /mirror/slackware-13.1 . The files for tftp will be on /tftpboot/slackware-13.1.

- Export the mirrror on NFS: Add `/mirror/Slackware 192.168.123.0/24(ro,sync,insecure,all_squash)'` to `/etc/exports`
- Create the mirror directory: `mkdir -p /mirror/slackware-13.1`
- `chmod +x /etc/rc.d/rc.nfsd`
- `/etc/rc.d/rc.nfsd restart`
- `mkdir /tftpboot`
- `mkdir /tftpboot/slackware-13.1`
- `mkdir /tftpboot/slackware-13.1/pxelinux.cfg`
- `cp /usr/share/syslinux/pxelinux.0 /tftpboot/slackware-13.1`
- `cp /mirror/slackware-13.1/isolinux/message.txt /tftpboot/slackware-13.1/`
- `cp /mirror/slackware-13.1/isolinux/f2.txt /tftpboot/slackware-13.1/`
- `cp -a /mirror/slackware-13.1/kernels /tftpboot/slackware-13.1/`
- `cp /mirror/slackware-13.1/usb-and-pxe-installers/pxelinux.cfg_default \ /tftpboot/slackware-13.1/pxelinux.cfg/default`
- `cp /mirror/slackware-13.1/isolinux/initrd.img /tftpboot/slackware-13.1/`

In order to use the dhcp, tftp, bootp protocols from dnsmasq, the file `/etc/dnsmasq.conf` should have the following options:

```
interface=eth0
interface=eth1
expand-hosts
domain=ssf.net
dhcp-range=192.168.123.2,192.168.123.250,255.255.255.0,12h
dhcp-option=vendor:PXEClient,1,0.0.0.0
dhcp-option-force=208,f1:00:74:7e
dhcp-option-force=210,/tftpboot/slackware-13.1/
dhcp-boot=pxelinux.0
enable-tftp
tftp-root=/tftpboot/slackware-13.1
```

3.2 Clientes

Each client should try to boot up from the net, and if the client is able to find the pxe server, all the installation will proceed without problem. When the slackware installer asks for the source of the files, you should configure it by using the option "Install from NFS". You will need to provide the ip address of the server (192.168.123.1) and the directory shared by NFS (/mirror/slackware-13.1) . After this, the installation will proceed as usual.

3.3 Post-instalación

Despues de la instalación se espera que todo funcione bien. En caso de ser necesario, el sonido se configura con el comando `alsaconf`, y el video con el comando `xorgconf` . En particular, el comando llamado `xorgsetup` configura practicamente todo de una manera fácil. Se recomienda ejecutarlo y reiniciar Algunas veces se presentan problemas con tarjetas de video como la ATI (por ejemplo). En ese caso, es aconsejbale usar el comando arriba mencionado, y luego editar el archivo de configuración /etc/X11/xorg.conf, cambiando el driver propietario por vesa y luego colocando la configuración virtual por subsección, por ejemplo,

```
Subsection "Display"
Viewport 0 0
Depth 24
Virtual 1289 1024
```

Para que el sistema inicie siempre en modo gráfico, debe editarse el archivo `/etc/inittab` y cambiar el runlevel 3 a runlevel 4, es decir, cambiar la línea `id:3:initdefault:` a `id:4:initdefault:` . Luego reiniciar.

Furthermore, you could also activate additional consoles in runlevel 4 by editing (adding a 4) to the respawn lines in the same file.

Furthermore, it is advisable to change the lilo time for each client to 4 minutes, while leaving the lilo time of the server to 5 secs or similar. To do this, edit the file `/etc/lilo.conf` and later use the command `lilo` .

4 Configuración de la red interna

4.1 SERVER

Se debe configurar a ETH1 con ip fijo igual a 192.168.123.1 , y se debe configurar a ETH0 para que haga DHCP, ojalá con ip fijo solicitado anteriormente, en este caso 168.176.34.36. La mejor manera de hacer lo anterior es usar primero la herramienta netconfig, respondiendo todas la preguntas con la info apropiada, y escogiendo yes para hacer dhcp. Luego, se debe editar el archivo /etc/rc.d/rc.inet1.conf y se deben verificar que ETH0 haga DHCP y que ETH1 tenga ip fijo. Cada sección debe lucir así:

```
# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_IPADDR[0]="168.176.34.36"
DHCP_HOSTNAME[0]=""

# Config information for eth1:
IPADDR[1]="192.168.123.1"
NETMASK[1]="255.255.255.0"
USE_DHCP[1]=""
DHCP_HOSTNAME[1]=""
```

Las otras secciones deben estar vacías "" o comentadas con #.

4.1.1 Dhcpc client customization

It would be great to run some commands just after eth0 has got the dhcp external IP. That could be done by writing the `/etc/dhcpc/dhcpc.sh`. For example, it can be used to add 192.168.123.1 as the nameserver BEFORE any other external dns server. See thi example file at the end.

Cuando el servidor de la red interna ejecuta dhcpc e la red externa, la lista de servidores DNS es reescrita por el servidor DHCP que responde a la solicitud. Pero es necesario que el server de la red interna sea uno de los servidores DNS, o de lo contrario no se podran resolver los nombres de la red interna. En esta sección se explica cómo lograrlo.

Edit the file `/etc/resolv.conf.head` . This file will be added at the beginning of the `resolv.conf` file. Put something like

```
nameserver 127.0.0.1
```

4.2 CLIENT

Cada cliente simplemente se debe configurar para que haga dhcp. Cada hostname puede configurarse al antojo, el servidor DHCP reconfigurará los nombres para que queden numerados en la red interna. Por lo tanto, el uso de netconfig, o la edición de /etc/rc.d/rc.inet1.conf son suficientes. El archivo /etc/rc.d/rc.inet1.conf debe lucir como

```
# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]=""
```

Las otras secciones deben estar vacías "" o comentadas con #.

Tal vez la respuesta del servidor sea muy lenta, más que los 10 segundos por defecto, de manera que una opción que puede servir es incrementar el timeout de dhcp, quedando la sección así:

```
# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]=""
DHCP_TIMEOUT[0]="30"
```

4.3 Test

Aún no se puede hacer testing dado que no se ha configurado completamente el servidor DHCP.

5 Añadir el ip a un servidor DNS dinámico externo para computadores lejanos dentro de la misma red de la universidad

5.1 SERVER

For dynamics dns we are going to use the service on www.dyndns.org. The name for the server will be `serverssf1.dyndns.org` . We have to download the ddclient for Linux and to build it. The easiest way is by using slackbuilds.org . The ddclient configuration goes on /etc/ddclient/ddclient.conf, and should have the following options

```
daemon=120                                  # check every 300 seconds
syslog=yes                                  # log update msgs to syslog
mail=root                                   # mail all msgs to root
mail-failure=root                           # mail failed update msgs to root
pid=/var/run/ddclient.pid                   # record PID in file.
#ssl=yes                                            # use ssl-support. Works with
use=if,                      if=eth0                # via interfaces
proxy=proxy2.unal.edu.co:8080                       # default proxy
login=dyndnsusername                              # default login
password=dyndnspassword                                     # default password
server=members.dyndns.org,                  \
protocol=dyndns2                            \
serverssf1.dyndns.org
```

5.2 TEST

nslookup serverssf1.dyndns.org

6 RED INTERNA: Configuración de DHCP, NAT, MASQUERADING, NTP, DNS, etc.

6.1 SERVER

La configuración de todos estos servicios se realiza facilmente con el paquete dnsmasq. Una vez instalado, el archivo /etc/dnsmasq.conf debe ser configurado para el caso particular. Se remite al lector a toda la documentación que acompaña a el paquete. El ejemplo del archivo dnsmasq.conf se encuentra al final de este documento. El servicio dnsmasq debe ser activado desde el inicio, es decir, /etc/rc.d/rc.dnsmasq debe ser ejecutable por root.

Las opciones importantes a configurar son:

- Interface en la que se escuchan los request de dhcp (ETH1 y ETH0):

```
# If you want dnsmasq to listen for DHCP and DNS requests only on
# specified interfaces (and the loopback) give the name of the
# interface (eg eth0) here.
# Repeat the line for more than one interface.
interface=eth1
interface=eth0
```

- Dominio:

```
domain=ssf.net
```

- Rango de dhcp (hay muchas formas de hacerlo, ver la documentación):

```
dhcp-range=192.168.123.2,192.168.123.250,255.255.255.0,12h
```

- Ignorar request de otras máquinas: Esta medida requiere conocer las macs de los clientes y SE ACONSEJA.

```
dhcp-host=*:*:*:*:*:*,ignore
```

- Asignar ips de acuerdo a la mac, tener en cuenta que los nosmbres de DNS serán los asignados por el DHCP:

```
dhcp-host=00:11:11:82:EB:26,ssf32-02,192.168.123.2
dhcp-host=00:12:3F:A7:28:C3,ssf32-03,192.168.123.3
dhcp-host=00:07:E9:F0:C4:C9,ssf32-04,192.168.123.4
dhcp-host=00:14:22:3B:24:F3,ssf64-01,192.168.123.5
```

  Y así sucesivamente.

- Opciones de acuerdo al RFC 2132 (google ?) en este caso la mascara de red, el gateway, winsservers y dns server.

```
dhcp-option=1,255.255.255.0
dhcp-option=6,192.168.123.1
dhcp-option=44,168.176.160.22,168.176.160.23
dhcp-option=41,192.168.123.1
```

- Ntp server:

```
# Set the NTP time server addresses to 192.168.0.4 and 10.10.0.5
#dhcp-option=option:ntp-server,192.168.0.4,10.10.0.5
dhcp-option=option:ntp-server,192.168.123.1
# Set the NTP time server address to be the same machine as
# is running dnsmasq
dhcp-option=42,0.0.0.0
```

- NIS domain (for user authentification)

  ```
  dhcp-option=40,ssfservernis
  ```

  Por último, es importante comentar la línea de 127.0.0.1 en el `/etc/hosts` y reemplazarla por el ip del servidor, 192.168.123.1 . El archivo debe quedar así:

  ```
  #
  # hosts         This file describes a number of hostname-to-address
  #               mappings for the TCP/IP subsystem.  It is mostly
  #               used at boot time, when no name servers are running.
  #               On small systems, this file can be used instead of a
  #               "named" name server.  Just add the names, addresses
  #               and any aliases to this file...
  #
  # By the way, Arnt Gulbrandsen <agulbra@nvg.unit.no> says that 127.0.0.1
  # should NEVER be named with the name of the machine.  It causes problems
  # for some (stupid) programs, irc and reputedly talk. :^)
  #

  # For loopbacking.
  127.0.0.1               localhost
  #127.0.0.1              ssf1.ssf.net ssf1
  192.168.123.1           ssf1.ssf.net ssf1

  # End of hosts.
  ```

6.2 TEST

Reiniciar el servidor, luego reiniciar los clientes. Desde un cliente hacer ping al server

```
ping 192.168.123.1
```

y debe ser exitoso, mostrando algo como

```
PING 192.168.123.1 (192.168.123.1) 56(84) bytes of data.
64 bytes from 192.168.123.1: icmp_seq=1 ttl=64 time=0.158 ms
64 bytes from 192.168.123.1: icmp_seq=2 ttl=64 time=0.147 ms
64 bytes from 192.168.123.1: icmp_seq=3 ttl=64 time=0.142 ms
64 bytes from 192.168.123.1: icmp_seq=4 ttl=64 time=0.145 ms

--- 192.168.123.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.142/0.148/0.158/0.006 ms
```

De lo contrario hay que encontrar el problema. Un error común es que dnsmasq no esté corriendo.

7 FIREWALL

Se usará al firewall de arno (Arno iptables, http://rocky.eld.leidenuniv.nl/). La versión actual es la 1.9. La configuración está en /etc/arno-iptables-firewall/firewall.conf. El firewall se instala en el server conectado a la red externa, se supone que la red interna es confiable.

El arno-iptables-firewall instala varios archivos y permite la creación del script rc.firewall . Leer la doc.

Para una instalación sencilla, usar el comando `install.sh` que viene en el paquete. Luego, linkear a `rc.firewall`, `ln -s /etc/init.d/arno-iptables-firewall /etc/rc.d/rc.firewall` .

La configuración queda en el archivo `/etc/arno-iptables-firewall/firewall.conf` . Revisarla, editar las rutas de los comandos `iptables` y `ip6tables`.

Add a hook to run the firewall each time the external interface is updated, by adding the following line

```
/etc/rc.d/rc.firewall restart
```

to the file `/etc/dhcpcd.exit-hook`

7.1 SERVER

Configurar el firewall.conf (ver ejemplo al final)

7.2 CLIENT

No se instala el firewall en los clientes, sólo en el servidor de red externo. Se supone que la red interna es confiable.

7.3 TEST

Verificar que una vez reiniciado el firewall, los computadores internos y el externo podrán navegar.

8 VPN : Computers outside the internal LAN but inside University LAN

In this section we are going to configure show how to setup a VPN, using openVPN, to connect computers outside the internal LAN to the main server in order to share NFS files and to allow NIS authentification. The network segment will be 10.88.55.0/24 .

Heavily based on the openVPN howto: `http://openvpn.net/howto.html`

8.1 SERVER

- Setting up your own Certificate Authority (CA) and generating certificates and keys for an OpenVPN server and multiple clients:

  1. Go to the directory /usr/share/doc/openvpn-2.0.9, where 2.0.9 is the version of openvpn. cp this directory to another one to save the new files without destroying the originals.
     `cp -a /usr/share/doc/openvpn-2.0.9 /usr/share/doc/openvpn-2.0.9-mod`
  2. Go to the new dir, and enter the subdir `easy-rsa`.
  3. Setup the vars in the file `vars`, we can use the defaults, and load the vars: `. ./vars`
  4. Clean any previous: `./clean-all`
  5. Build the certificate: `./build-ca` , you can use default answers for everything but `Common Name` , where you should put something to describe your vpn network, like OpenVPN-SSF.
  6. Generate certificate and key for server: `./build-key-server ssfvpnserver` , you can use default parameters, BUT `Common Name` should be set to ssfvpnserver. Answer y for Sign the certificate and y to commit.
  7. Generate certificates and keys for the clients:
     `./build-key client1`
     `./build-key client2`
     `./build-key client3`
     MAKE SURE TO PUT THE NAME OF THE CLIENT IN `Common Name` : First one is client1, second one is client2, etc.
  8. Generate Diffie Hellman parameters: `./build-dh`
  9. Generate shared key for encrypted traffic with TLS: `openvpn --genkey --secret ta.key`

- You will have to copy the files to `/etc/openvpn/certs` and `/etc/openvpn/keys` :

  - ca.crt to server and clients, to dir /etc/openvpn/certs
  - ta.key to server and clients, to dir /etc/openvpn/keys
  - ca.key to server only, to dir /etc/openvpn/keys.
  - dh1024.pem to server only, to dir /etc/openvpn/.
  - server.crt to server only, to dir /etc/openvpn/certs.
  - server.key to server only, to dir /etc/openvpn/keys.
  - client1.crt to client1 only, to dir /etc/openvpn/certs.
  - client1.key to client1 only, to dir /etc/openvpn/keys.
  - client2.crt to client2 only, to dir /etc/openvpn/certs.
  - client2.key to client2 only, to dir /etc/openvpn/keys.
  - ⋮

- The server configuration file in /etc/openvpn/openvpn.conf should have the following options:

```
cd /etc/openvpn
proto udp
port 1194
verb 3
log-append /var/log/openvpn.log
daemon
dev tun
persist-tun
persist-key
server 10.88.55.0 255.255.255.0
ifconfig-pool-persist ips.txt
client-to-client
cipher BF-CBC
ca certs/ca.crt
dh dh1024.pem
cert certs/ssfvpnserver.crt
key keys/ssfvpnserver.key
tls-auth keys/ta.key 0
```

To create the device automatically, you should load the tun module: modprobe tun .

You must open the port 1194 in the firewall OPEN_UDP="53,67,1194", and add the interface tun0 to the fully trsuted interfaces TRUSTED_IF="tun0"

8.2 CLIENT(S)

Each client configuration should have the following options (/etc/openvpn/openvpn.conf):

```
client
cd /etc/openvpn
remote serverssf1.dyndns.org 1194
proto udp
port 1194
verb 3
log-append /var/log/openvpn.log
dev tun
persist-tun
persist-key
cipher BF-CBC
```

```
ca certs/ca.crt
cert certs/ssfvpnclient1.crt
key keys/ssfvpnclient1.key
tls-auth keys/ta.key 1
```

Change ssfvpnclient1 for the actual name of the client in the vpn.

8.3 TEST

- Start the server : `openvpn --config /etc/openvpn/openvpn.conf`
  Something like the following should appear in `/var/log/openvpn.log`:

```
Wed Apr 14 23:14:22 2010 OpenVPN 2.0.9 i486-slackware-linux [SSL] [LZO] [EPOLL] built on Jun 11 2007
Wed Apr 14 23:14:22 2010 WARNING: --keepalive option is missing from server config
Wed Apr 14 23:14:22 2010 Diffie-Hellman initialized with 1024 bit key
Wed Apr 14 23:14:22 2010 Control Channel Authentication: using 'keys/ta.key' as a OpenVPN static key fi
Wed Apr 14 23:14:22 2010 Outgoing Control Channel Authentication: Using 160 bit message hash 'SHA1' for
Wed Apr 14 23:14:22 2010 Incoming Control Channel Authentication: Using 160 bit message hash 'SHA1' for
Wed Apr 14 23:14:22 2010 TLS-Auth MTU parms [ L:1541 D:166 EF:66 EB:0 ET:0 EL:0 ]
Wed Apr 14 23:14:22 2010 TUN/TAP device tun0 opened
Wed Apr 14 23:14:22 2010 /sbin/ifconfig tun0 10.88.55.1 pointopoint 10.88.55.2 mtu 1500
Wed Apr 14 23:14:22 2010 /sbin/route add -net 10.88.55.0 netmask 255.255.255.0 gw 10.88.55.2
Wed Apr 14 23:14:22 2010 Data Channel MTU parms [ L:1541 D:1450 EF:41 EB:4 ET:0 EL:0 ]
Wed Apr 14 23:14:22 2010 UDPv4 link local (bound): [undef]:1194
Wed Apr 14 23:14:22 2010 UDPv4 link remote: [undef]
Wed Apr 14 23:14:22 2010 MULTI: multi_init called, r=256 v=256
Wed Apr 14 23:14:22 2010 IFCONFIG POOL: base=10.88.55.4 size=62
Wed Apr 14 23:14:22 2010 IFCONFIG POOL LIST
Wed Apr 14 23:14:22 2010 ssfvpnclient2,10.88.55.4
Wed Apr 14 23:14:22 2010 ssfvpnclient1,10.88.55.8
Wed Apr 14 23:14:22 2010 Initialization Sequence Completed
```

- Start each client (forst make a `modprobe tun`): `openvpn --config /etc/openvpn/openvpn.conf`
  Something like the following should appear in `/var/log/openvpn.log`:

```
Wed Apr 14 23:19:48 2010 OpenVPN 2.0.9 i486-slackware-linux [SSL] [LZO] [EPOLL] built on Jun 11 2007
Wed Apr 14 23:19:48 2010 WARNING: No server certificate verification method has been enabled.  See http
Wed Apr 14 23:19:48 2010 Control Channel Authentication: using 'keys/ta.key' as a OpenVPN static key fi
Wed Apr 14 23:19:48 2010 Outgoing Control Channel Authentication: Using 160 bit message hash 'SHA1' for
Wed Apr 14 23:19:48 2010 Incoming Control Channel Authentication: Using 160 bit message hash 'SHA1' for
Wed Apr 14 23:19:48 2010 Control Channel MTU parms [ L:1541 D:166 EF:66 EB:0 ET:0 EL:0 ]
Wed Apr 14 23:19:48 2010 Data Channel MTU parms [ L:1541 D:1450 EF:41 EB:4 ET:0 EL:0 ]
Wed Apr 14 23:19:48 2010 Local Options hash (VER=V4): '70f5b3af'
Wed Apr 14 23:19:48 2010 Expected Remote Options hash (VER=V4): 'a2e2498c'
Wed Apr 14 23:19:48 2010 UDPv4 link local (bound): [undef]:1194
Wed Apr 14 23:19:48 2010 UDPv4 link remote: 168.176.34.36:1194
Wed Apr 14 23:19:48 2010 TLS: Initial packet from 168.176.34.36:1194, sid=4df00a24 3d78a650
Wed Apr 14 23:19:48 2010 VERIFY OK: depth=1, /C=KG/ST=NA/L=BISHKEK/O=OpenVPN-TEST/CN=OpenVPN-SSF/emailA
Wed Apr 14 23:19:48 2010 VERIFY OK: depth=0, /C=KG/ST=NA/O=OpenVPN-TEST/CN=ssfvpnserver/emailAddress=me
Wed Apr 14 23:19:48 2010 Data Channel Encrypt: Cipher 'BF-CBC' initialized with 128 bit key
Wed Apr 14 23:19:48 2010 Data Channel Encrypt: Using 160 bit message hash 'SHA1' for HMAC authenticatio
Wed Apr 14 23:19:48 2010 Data Channel Decrypt: Cipher 'BF-CBC' initialized with 128 bit key
Wed Apr 14 23:19:48 2010 Data Channel Decrypt: Using 160 bit message hash 'SHA1' for HMAC authenticatio
Wed Apr 14 23:19:48 2010 Control Channel: TLSv1, cipher TLSv1/SSLv3 DHE-RSA-AES256-SHA, 1024 bit RSA
Wed Apr 14 23:19:48 2010 [ssfvpnserver] Peer Connection Initiated with 168.176.34.36:1194
```

```
Wed Apr 14 23:19:49 2010 SENT CONTROL [ssfvpnserver]: 'PUSH_REQUEST' (status=1)
Wed Apr 14 23:19:49 2010 PUSH: Received control message: 'PUSH_REPLY,route 10.88.55.0 255.255.255.0,ifc
Wed Apr 14 23:19:49 2010 OPTIONS IMPORT: --ifconfig/up options modified
Wed Apr 14 23:19:49 2010 OPTIONS IMPORT: route options modified
Wed Apr 14 23:19:49 2010 TUN/TAP device tun0 opened
Wed Apr 14 23:19:49 2010 /sbin/ifconfig tun0 10.88.55.6 pointopoint 10.88.55.5 mtu 1500
Wed Apr 14 23:19:49 2010 /sbin/route add -net 10.88.55.0 netmask 255.255.255.0 gw 10.88.55.5
Wed Apr 14 23:19:49 2010 Initialization Sequence Completed
```

Now you could ping the server from the client: `ping 10.88.55.1` .

NOTE: You will have to add the network 10.88.55.0/24 to the trusted networks for nfs and nis.

On the server, the file /etc/openvpn/ips.txt can be used to configure the ips for each client.

DO NOT FORGET TO START THE VPN TUNNEL AT THE BEGINNING

9 Comandos de bash múltiples máquinas (paralelos)

Esta sección está diseñada para facilitar la edición y ejecución de múltiples instrucciones en múltiples computadores al mismo tiempo. Se utilizarán las herramientas para manejo de clusters C3 (http://www.csm.ornl.gov/torc/C3/).

DNS or hostname resolution must be supported and working.

9.1 SERVER

Seguir las instrucciones del archivo INSTALL.

- Uncompress the package and enter the directory.
- If you want to install the tools in a directory different from the default `/opt/c3-4` , replace `basedir` in the script `Install-c3` .
- Install the tools by executing the install script: `# bash Install-c3`
- Configuration: the clusters are written on /etc/c3.conf . Example :

```
cluster local {
        ssf1:ssf1 # head node
        ssf1
        ssf[2-3]
        10.88.55.6
        10.88.55.10
}
```

  The last ones are for the vpn, until now is not linked with some dns mechanism.

- For each c3 command make a link to `/usr/local/bin` :

```
for a in /opt/c3-4/*; do if [ -x $a ]; then echo $a; \
ln -s $a /usr/local/bin/; fi ; done
```

- `ln -s /usr/bin/python /usr/local/bin/python2`

Until we have configured succesfully the c3 tools, but each time we use the command we are prommted for a password, so we need to configure automatic ssh login. How to do that? Google! (the range 1-2 should be changed to the actual number of nodes)

- `ssh-keygen -t dsa`
- `chmod 700 ~/.ssh/`

- (Not needed when the directory .ssh is shared b NFS, i.e. a typical user)
  `cpush ~/.ssh/id_dsa.pub ~/`
- `cexec :1-3 "mkdir .ssh"`
- `cexec :1-3 "cat id_dsa.pub >> .ssh/authorized_keys"`
- `DONE!`

9.2 TEST

`cexec ls -l` should work.

10 Configuración del NFS

El NFS permite compartir archivos y directiorios de forma trasnparente en la red. En este caso, suponemos que el server va a exportar el directorio /home a los clientes.

10.1 SERVER

Verificar que `/etc/rc.d/rc.nfs` sea ejecutable.

Se debe editar el archivo /etc/exports . En este archivo se especifica el directorio a compartir, a quién se le comparte (segmento de red) y las opciones. LEER EL MANUAL O HOWTO. Un ejemplo puede ser

```
/home
192.168.123.0/255.255.255.0(fsid=0,rw,sync,no_root_squash,no_subtree_check)  \
10.88.55.0/255.255.255.0(fsid=0,rw,sync,no_root_squash,no_subtree_check)
```

Adicionalmente, es necesario editar los archivos /etc/hosts.allow y /etc/hosts.deny, que están relacionados con la seguridad y con la activación del servicio RPC, necesario para NFS. En host.deny debe aparecer

```
portmap:ALL
lockd:ALL
mountd:ALL
rquotad:ALL
statd:ALL
```

En el archivo hosts.allow se otorga la autorizacion a la red en cuestión,

```
portmap: 192.168.123.0/255.255.255.0 , 10.88.55.0/255.255.255.0
lockd: 192.168.123.0/255.255.255.0 , 10.88.55.0/255.255.255.0
mountd: 192.168.123.0/255.255.255.0  , 10.88.55.0/255.255.255.0
rquotad: 192.168.123.0/255.255.255.0  , 10.88.55.0/255.255.255.0
statd: 192.168.123.0/255.255.255.0 , 10.88.55.0/255.255.255.0
```

10.2 CLIENT

Se debe editar el archivo de montaje de directorios, el /etc/fstab. For internal clients:

```
192.168.123.1:/home     /home   nfs     rw,hard,intr    0       0
192.168.123.1:/mnt/NFS/PACKAGES /mnt/NFS/PACKAGES       nfs     rw,hard,intr    0       0
```

or, for vpn external clients,

```
10.88.55.1:/home     /home   nfs     rw,hard,intr    0       0
10.88.55.1:/mnt/NFS/PACKAGES /mnt/NFS/PACKAGES       nfs     rw,hard,intr    0       0
```

10.3 TEST

Reiniciar los equipos. En el servidor, el comando rpcinfo -p debe mostrar, entre otros, que los servicios portmapper y demás están activos. Manualmente se pueden reinicar los servicios sin reiniciar el sistema escribiendo

`/etc/rc.d/rc.inet2 restart`

En este momento todos los equipos deben ver el directorio compartido y pueden trabajar sobre el mismo.

11 NIS

Este servicio controla la autentificación de los usuarios. Otra alternativa puede ser LDAP, pero no se tratará acá.
El setup de NIS es muy tricky, por favor remitirse al howto todo el tiempo.

11.1 SERVER

- Configurar el dominio nis:

  ```
  echo domain_name > /etc/defaultdomain
  ```

  donde `domain_name` se refiere, en este caso, a `ssfservernis` .

- Configurar la base de datos de usuarios (quienes ya deben existir).

  ```
  # cd /var/yp
  (editar Makefile: MERGE_PASSWD = true -> false)
  (editar Makefile: MERGE_GROUP = true -> false)
  (editar Makefile: MINUID=500 -> pasar al valor apropiado (generalmente
  asi esta bien))
  (editar Makefile: MINGID=500 -> pasar al valor apropiado, de manera
  que plugdev y demas se pasen por NIS, puede ser 1)
  (comentar plubickey an el target all:)
  # ypserv
  # make
  ```

  La configuracion del master y de master slaves puede hacerse con el comando `/usr/lib/yp/ypinit -m`

- Configurar el script de inicio de NIS:

  ```
  #chmod +x /etc/rc.d/rc.yp
  (editar rc.yp, descomentar lo referente a defaultdomain)
  (editar rc.yp, descomentar lo referente a ypserv)
  (editar rc.yp, descomentar lo referente a yppasswdd)
  ```

11.2 CLIENT

- Domain name :

  ```
  echo domain_name > /etc/defaultdomain
  ```

- Editar /etc/rc.d/rc.yp y descomentar lo concerniente a NIS client. Descomentar el cat del default domain. Descomentar el yp_bind -broadcast SOLO PARA LOS PCS INTERNOS. Para los pcs en la LAN unidos por vpn, descomentar lo mismo pero eliminar el -broadcast, y en el archivo `/etc/yp.conf` anhadir la línea `ypserver 10.88.55.1`.

- Editar /etc/nsswitch.conf:

```
Comentar passwd: compact
Comentar group: compact
Descomentar passwd: files nis
Descomentar shadow: files nis
Descomentar group: files nis
```

- Añadir "+:: ..." (el número de : depende del archivo) al final de los archivos correspondientes:

```
echo +:::::: >> /etc/passwd
echo +::::::: >> /etc/shadow
echo +::: >> /etc/group
echo +:: ... >> /etc/gshadow
```

11.3 TEST

REINICIAR EL SERVER Y LOS CLIENTES y verificar que un usuario en el server puede loguearse exitosamente en otro computador de la red.

12 BACKUP

Para el backup se usa el sistema simple de backup de la página http://foc.neoartis.org/progs/backup/ , http://freshmeat.net/projects/loopbackup/ .

Supongo que el cliente 192.168.123.7 es el servidor de backup.

Para este caso se aconseja actualizar (mediante cron) el backup cada 2 semanas. Se guardarn tres copias, es decir, un mes y medio.

12.1 SERVER

12.2 CLIENT

Para configurarlo, simplemente seguir las instrucciones de la página y/o paquete.

El cron debe ser configurado en la misma máquina mismo, por ejemplo usando kcron. Si se prefiere `crontab -e`, la línea sería

```
0 7 1,15 * * /usr/local/backup/bakcup.sh 1>/dev/null 2>/dev/null
```

13 Recrear la base de datos a partir de una ya existente

13.1 SERVER

Reiniciar los servicios NIS mediante `/etc/rc.d/rc.inet2 restart`

13.2 CLIENT

Reiniciar los servicios NIS mediante `/etc/rc.d/rc.inet2 restart`

13.3 TEST

Probra logueandose con diferentes cuentas en diferentes computadores.

14 QUOTA

Toda la información se encuentra en el Quota-Howto. El procedimiento es sencillo: Asignar las cuotas, configurar el fstab para asignarle al sistema de archivos el soporte para cuota, asignar las cuotas a cada usuario, reiniciar. Las cuotas son compatibles con NFS. En este caso, haremos que la cuota sea de 1.4G.

Journaled quita will be configured.

14.1 SERVER

Modificar fstab, añadiendo el soporte para quota en las opciones.

```
/dev/sda5        /home            reiserfs    defaults,usrquota,usrjquota=aquota.user,jqfmt=vfsv          1
```

```
touch /home/aquota.user
chmod 600 /aquota.*
mount -o remount /home
```

Añadir línea en crontab para verificar semanalmente la cuota:

```
0 3 * * 0 /sbin/quotacheck -avug
```

Primero editar la cuota para un usuario típico, digamos tmpuser.

```
export EDITOR=emacs
edquota -u tmpuser
```

Setear el soft de los blocks en 3000000, y el hard en 3200000. El grace time típico es de 7 días.

Ahora editar la cuota para los demás usuarios usando a tmpuser como prototipo.

```
for a in $(cat /etc/passwd | awk '{ FS=":";if ($3 > 1000) print $1}'); \
do edquota -p test $a ;done
```

Por último, se puede editar la cuota de usuarios particulares para que tengan menos o más quota.

15 Directorio NFS para paquetes

En este caso tendremos un directorio en el servidor principal y que será compartido con los pcs de la red interna por medio del NFS (que se supone ya configurado). Este directorio está orientado a mantener una lista de pauqtes comunes a instlar en todos los sistemas.

15.1 SERVER

- Crear el directorio: `mkdirhier /mnt/NFS/PACKAGES`
- Exportar el directorio por NFS: Incluir la siguiente línea en el `/etc/exports` :
  `/mnt/NFS/PACKAGES 192.168.123.0/255.255.255.0(rw,sync,no_root_squash)`
- Reexportar: `/etc/rc.d/rc.inet2 restart`

15.2 CLIENT

- Añadir la siguiente línea a el `/etc/fstab` :
  `192.168.123.1:/mnt/NFS/PACKAGES /mnt/NFS/PACKAGES nfs \ rw,hard,intr 0 0`
- Crear el directorio: `mkdirhier /mnt/NFS/PACKAGES`
- Remontar el NFS: `/etc/rc.d/rc.inet2 restart`

16 Impresora

El SSF cuenta con una impresora que tiene su propia tarjeta de red. El ip asignado a la impresora es 192.168.123.9 . En este caso se configurará CUPS para que "hable" con esa impresora, una HP Laserjet 2420dn.

16.1 SERVER

- Add the network printer via the cups interface on localhost:631
- Backup the cups directory on each slave:
  `cexec :1-2 "cp -a /etc/cups ~/cupsOLD"`
- Copy (overwrite) the server cups directory to the slaves and restart:
  ```
  tar czf cups.tar.gz /etc/cups
  cpush :1-2 cups.tar.gz cupsHEAD.tar.gz
  cexec :1-2 "tar xzf cupsHEAD.tar.gz"
  cexec :1-2 "rsync -av etc/cups/ /etc/cups/"
  cexec :1-2 "/etc/rc.d/rc.cups restart"
  ```

16.2 TEST

Print test pages on each computer

17 Upgrade slackware to latest patches with slackpkg

- Uncomment one mirror in `/etc/slackpkg/mirrors` (I use to use tds mirrors), on each computer
- Upgrade slackpkg : `cexec slackpkg update`
  If you are behind a firewall, export the appropiate value of the http_proxy variable
- Upgrade the patched programs : `cexec slackpkg upgrade patches`

18 Software adicional

El siguiente software es sugerido para tener en todos los cmputadores. Lo ideal sería dedicar un directorio para los paquetes y distribuirlo por NFS para que los demás pc los vean y puedan instalarlos. Compilarlos una sola vez. (Verificar si se puede correr en paralelo el comandod e instalación o si slaptget puede hacer algo al respecto.)

- flash-player-plugin
- eigen
- boost
- open office.
- xmgrace: Plotting Utility
- aterm: Great terminal, lightweight
- djvulibre: Djvu stuff
- lame: Mp3 encoding
- xtail: For x tail messages
- kile

- unrar
- jdk
- acroread
- *** Multimedia *** mplayer: Obviously
- fileligth: Hard disk usage tool
- htop
- skype
- povray
- pdftk: for hacking pdf files
- blender: 3d content creation suite
- valgrind: Award winning tool for debugging and profiling
- blas, cblas, lapack, openmpi, hdf5, libctl, harminv, h5utils, nlopt, fftw V2 (old for mpb), meep(MIT), mpb(MIT), camfr
- mechsys: MTL, voro++, tetgen, metis, suitesparse, mechsys

19 Scanner

The scanner at the office is a Benq 4300 old scanner. To make it work, a firmware is needed, look at the information related to sane and snapscan backend, `http://snapscan.sourceforge.net/` . The scanner is a Acer / Benq 4300 FlatbedScanner23, something that you can know by running the commands `sane-find-scanner` and, if successfull, `scanimage -L` .

Fortunately, SLackware 13.1 detects successfully the Scanner!

They say it is a `FlatbedScanner21`, but the working firmware is for the `FlatbedScanner23` From this information, we know we need the firmware `u176v046.bin`. It is quite tricky to find this file, you can google and/or try to download it from the vendor site. One link is `http://outlands.ca/linux/snapscan-firmware.html` `http://ubuntu-col.blogspot.com/2009/03/ubuntu-8_18.html` . After downloading, copy the firmware `u176v046.bin` to the folder `/usr/share/sane/snapscan/`, open the file `/etc/sane.d/snapscan.conf` and put the full path to the firmware. Unplug and unplug the scanner. Done. You can scan by using xsane or gimp or xscanimage.

The users will to be in the group scanner. Restart the messagebus `/etc/rc.d/rc.messagebus restart`

To share the scanner on the network: `http://tldp.org/HOWTO/Scanner-HOWTO/sane.html#CONFIG-SANE`

19.1 SERVER (PC with the scanner)

```
if ! id saned; then
 groupadd saned;
 useradd -g saned -G scanner -s /bin/false -d /dev/null saned;
fi
```

Add the network range in `/etc/sane.d/saned.conf`

Add the saned service, `sane 6566/tcp` , to `/etc/services`

Add

```
sane-port stream tcp nowait saned.saned /usr/sbin/tcpd/usr/sbin/saned
```

to `/etc/inetd.conf`

Add ALL:ALL to /etc/hosts.allow

Restart inetd

19.2 CLIENT

- Add ip address of scanner server to the file `/etc/sane.d/net.conf`

20 Configuración de PHP en el server

20.1 SERVER

- En el archivo `/etc/httpd/httpd.conf` descomentar la línea `#Include /etc/httpd/mod_php.conf`
- En el mismo archivo, añadir
  `AddType application/x-httpd-php .php .php3 .phtml`
  `AddType application/x-httpd-php-source .phps`
  en el módulo `<IfModule mime_module>`.
- Reiniciar el servicio httpd.

20.2 TEST

Create a document called, for example, `info.php`, with the following content

```
<html>
<head>
<title> PHP Test Script </title>
</head>
<body>
<?php
phpinfo( );
?>
</body>
</html>
```

Save this document on `/srv/www/htdocs/`, and then open a browser and go to the URL `localhost/info.php` . You should see the info of the php package.

21 Comandos sudo

- Permitir que los usuarios ejecuten comandos de root: Escribir los comandos en el sudoers.
  `cexec :0-2 "echo '%users  ALL=/etc/rc.d/rc.inet1' >> /etc/sudoers"`
  `cexec :0-2 "echo '%users  ALL=/etc/rc.d/rc.inet2' >> /etc/sudoers"`
  `cexec :0-2 "echo '%users  ALL=/etc/rc.d/rc.cups' >> /etc/sudoers"`

22 Replacing NIS woth KerberosV5

Install Kerberos, or Heimdal if you prefer. We are going to follow the kerberos infraestructure howto and the Slackbuild, prefix /usr/, state /var.. See `http://aput.net/~jheiss/krbldap/howto.html`, `http://www.ornl.gov/~jar/HowToKerb.html`, `https://help.ubuntu.com/8.10/serverguide/C/kerberos.html`
Configuration:

```
./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var \
--disable-static \
--program-prefix= \
--program-suffix= \
```

22.1 SERVER

- Edit /etc/krb5.conf by following the examples in /usr/share/examples/krb5/krb5.conf.

```
[logging]
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmin.log
    default = FILE:/var/log/krb5lib.log

[libdefaults]
#    ticket_lifetime = 24000
       default_realm = SSF.NET
#        krb5_config = /usr/kerberos/lib/krb.conf
#        krb5_realms = /usr/kerberos/lib/krb.realms



[realms]
SSF.NET = {
kdc = ssf1.ssf.net:88
kdc = ssf7.ssf.net:88
admin_server = ssf1.ssf.net:749
default_domain = ssf.net
}

[domain_realm]
.ssf.net = SSF.NET
ssf.net = SSF.NET

[kdc]
   profile = /var/krb5kdc/kdc.conf
```

- Edit the /var/krb5kdc/kdc.conf file

```
[kdcdefaults]
        acl_file = /var/krb5kdc/kadm5.acl
        dict_file = /usr/share/dict/words
        admin_keytab = /var/krb5kdc/kadm5.keytab


[realms]
        SSF.NET = {
                database_name = /var/krb5kdc/principal
                admin_keytab = FILE:/var/krb5kdc/kadm5.keytab
                acl_file = /var/krb5kdc/kadm5.acl
                dict_file = /usr/share/dict/words
                key_stash_file = /var/krb5kdc/.k5.SSF.NET
                kdc_ports = 750,88
                #master_key_type = des3-hmac-sha1
                #supported_enctypes = des3-hmac-sha1:normal des-cbc-crc:normal
                #max_life = 10h 0m 0s
                #max_renewable_life = 7d 0h 0m 0s
        }
```

- Initialize the kerberos database:
  `kdb5_util create -s`
  You will be prompted for a KDC master password, dont forget it!
- `ln -s /var /usr/local`
- Edit `/var/krb5kdc/kadm5.acl` :

  ```
  */admin@SSF.NET   *
  ```

- Add admin users (principal users),
  `/usr/sbin/kadmin.local -q "addprinc username/admin"`
  , where username is a given username, for example admin or root.
- Add kadmin to start automatically on each reboot:
  `echo "/usr/sbin/kadmind &" >> /etc/rc.d/rc.local`
  `echo "/usr/sbin/krb5kdc &" >> /etc/rc.d/rc.local`
- Create kadmin keytabs:
  /usr/sbin/kadmin.local -q \
  "ktadd -k /var/krb5kdc/kadm5.keytab kadmin/admin kadmin/changepw"
- Start the daemons for testing. kinit username. klist
- Add the folloowing to /etc/dnsmasq.conf in order to be able to combine dnsmasq and kerberos

  ```
  # kerberos stuff
  srv-host=_kerberos._udp.SSF.NET,ssf1.ssf.net,88,1,0
  srv-host=_kerberos._tcp.SSF.NET,ssf1.ssf.net,88,1,0
  srv-host=_kerberos._udp.SSF.NET,ssf7.ssf.net,88,10,0
  srv-host=_kerberos._tcp.SSF.NET,ssf7.ssf.net,88,10,0
  srv-host=_kerberos-adm._tcp.SSF.NET,ssf1.ssf.net,749,1,0
  srv-host=_kpasswd._udp.SSF.NET,ssf1.ssf.net,464,1,0
  ```

  and restart dnsmasq. In this case, ssf7 will be the replication server.

22.2 Replication server

- Server: Create the host keys:
  `kadmin.local -q "addprinc -randkey host/ssf1.ssf.net"`
  `kadmin.local -q "addprinc -randkey host/ssf7.ssf.net"`
- Server: Extract the keytabs
  `kadmin.local -q "ktadd host/ssf1.ssf.net"`
  `kadmin.local -q "ktadd host/ssf7.ssf.net"`
- Server: Create /var/krb5kdc/kpropd.acl with
  `host/ssf1.ssf.net@SSF.NET`
  `host/ssf7.ssf.net@SSF.NET`
- Slave: Create an empty database
  `kdb5_util create -r SSF.NET -s`
- From server to slave, copy /etc/krb5.conf, /etc/krb5.keytab, /var/krb5kdc/kpropd.acl
- Server: Dump the database:
  `/usr/sbin/kdb5_util dump /usr/local/var/krb5kdc/slave_datatrans`

- Slave: Run the kpropd daemon standalone:
  `/usr/sbin/kpropd -S`
  and add that command to system initialization.
- Server: Propagate the database: `/usr/sbin/kprop -d -f /usr/local/var/krb5kdc/slave_datatrans ssf7.ssf.net`
  you should see the message "SUCCEEDED".
- Server: Create a script for sincronization /usr/local/bin/krb5prop.sh:

```
#!/bin/sh

/usr/sbin/kdb5_util dump /var/krb5kdc/slave_datatrans
/usr/sbin/kprop -f /var/krb5kdc/slave_datatrans ssf7.ssf.net > /dev/null
```

- Crontab for sincronization
  `15 * * * * /usr/local/bin/krb5prop.sh 1>/dev/null 2>/dev/null`

22.3 CLIENT

23 Archivos de configuración

23.1 /etc/rc.d/rc.inet1.conf

Server

```
# /etc/rc.d/rc.inet1.conf
#
# This file contains the configuration settings for network interfaces.
# If USE_DHCP[interface] is set to "yes", this overrides any other settings.
# If you don't have an interface, leave the settings null ("").

# You can configure network interfaces other than eth0,eth1... by setting
# IFNAME[interface] to the interface's name. If IFNAME[interface] is unset
# or empty, it is assumed you're configuring eth<interface>.

# Several other parameters are available, the end of this file contains a
# comprehensive set of examples.

# ============================================================================

# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]=""

# Config information for eth1:
IPADDR[1]="192.168.123.1"
NETMASK[1]="255.255.255.0"
USE_DHCP[1]=""
DHCP_HOSTNAME[1]=""

# Config information for eth2:
IPADDR[2]=""
NETMASK[2]=""
USE_DHCP[2]=""
DHCP_HOSTNAME[2]=""
```

```
# Config information for eth3:
IPADDR[3]=""
NETMASK[3]=""
USE_DHCP[3]=""
DHCP_HOSTNAME[3]=""

# Default gateway IP address:
GATEWAY=""

# Change this to "yes" for debugging output to stdout.  Unfortunately,
# /sbin/hotplug seems to disable stdout so you'll only see debugging output
# when rc.inet1 is called directly.
DEBUG_ETH_UP="no"

## Example config information for wlan0.  Uncomment the lines you need and fill
## in your info.  (You may not need all of these for your wireless network)
#IFNAME[4]="wlan0"
#IPADDR[4]=""
#NETMASK[4]=""
#USE_DHCP[4]="yes"
#DHCP_HOSTNAME[4]="icculus-wireless"
#DHCP_KEEPRESOLV[4]="yes"
#DHCP_KEEPNTP[4]="yes"
#DHCP_KEEPGW[4]="yes"
#DHCP_IPADDR[4]=""
#WLAN_ESSID[4]=BARRIER05
#WLAN_MODE[4]=Managed
##WLAN_RATE[4]="54M auto"
##WLAN_CHANNEL[4]="auto"
##WLAN_KEY[4]="D5AD1F04ACF048EC2D0B1C80C7"
##WLAN_IWPRIV[4]="set AuthMode=WPAPSK | set EncrypType=TKIP | set WPAPSK=96389dc66eaf7e6efd5b5523ae43c7925ff4df2f8b70994
#WLAN_WPA[4]="wpa_supplicant"
#WLAN_WPADRIVER[4]="ndiswrapper"

## Some examples of additional network parameters that you can use.
## Config information for wlan0:
#IFNAME[4]="wlan0"              # Use a different interface name nstead of
                                # the default 'eth4'
#HWADDR[4]="00:01:23:45:67:89"  # Overrule the card's hardware MAC address
#MTU[4]=""                      # The default MTU is 1500, but you might need
                                # 1360 when you use NAT'ed IPSec traffic.
#DHCP_KEEPRESOLV[4]="yes"       # If you dont want /etc/resolv.conf overwritten
#DHCP_KEEPNTP[4]="yes"          # If you don't want ntp.conf overwritten
#DHCP_KEEPGW[4]="yes"           # If you don't want the DHCP server to change
                                # your default gateway
#DHCP_IPADDR[4]=""              # Request a specific IP address from the DHCP
                                # server
#WLAN_ESSID[4]=DARKSTAR         # Here, you can override _any_ parameter
                                # defined in rc.wireless.conf, by prepending
                                # 'WLAN_' to the parameter's name. Useful for
                                # those with multiple wireless interfaces.
#WLAN_IWPRIV[4]="set AuthMode=WPAPSK | set EncrypType=TKIP | set WPAPSK=thekey"
                                # Some drivers require a private ioctl to be
                                # set through the iwpriv command. If more than
                                # one is required, you can place them in the
                                # IWPRIV parameter (separated with the pipe (|)
                                # character, see the example).
```

Client

```
# /etc/rc.d/rc.inet1.conf
#
# This file contains the configuration settings for network interfaces.
# If USE_DHCP[interface] is set to "yes", this overrides any other settings.
# If you don't have an interface, leave the settings null ("").

# You can configure network interfaces other than eth0,eth1... by setting
# IFNAME[interface] to the interface's name. If IFNAME[interface] is unset
# or empty, it is assumed you're configuring eth<interface>.

# Several other parameters are available, the end of this file contains a
# comprehensive set of examples.

# ============================================================================

# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]="192.168.123.1"

# Config information for eth1:
IPADDR[1]=""
NETMASK[1]=""
USE_DHCP[1]=""
DHCP_HOSTNAME[1]=""

# Config information for eth2:
IPADDR[2]=""
NETMASK[2]=""
USE_DHCP[2]=""
DHCP_HOSTNAME[2]=""

# Config information for eth3:
IPADDR[3]=""
NETMASK[3]=""
USE_DHCP[3]=""
DHCP_HOSTNAME[3]=""

# Default gateway IP address:
GATEWAY=""

# Change this to "yes" for debugging output to stdout.  Unfortunately,
# /sbin/hotplug seems to disable stdout so you'll only see debugging output
# when rc.inet1 is called directly.
DEBUG_ETH_UP="no"

## Example config information for wlan0.  Uncomment the lines you need and fill
## in your info.  (You may not need all of these for your wireless network)
#IFNAME[4]="wlan0"
#IPADDR[4]=""
#NETMASK[4]=""
#USE_DHCP[4]="yes"
#DHCP_HOSTNAME[4]="icculus-wireless"
#DHCP_KEEPRESOLV[4]="yes"
```

```
#DHCP_KEEPNTP[4]="yes"
#DHCP_KEEPGW[4]="yes"
#DHCP_IPADDR[4]=""
#WLAN_ESSID[4]=BARRIER05
#WLAN_MODE[4]=Managed
##WLAN_RATE[4]="54M auto"
##WLAN_CHANNEL[4]="auto"
##WLAN_KEY[4]="D5AD1F04ACF048EC2D0B1C80C7"
##WLAN_IWPRIV[4]="set AuthMode=WPAPSK | set EncrypType=TKIP | set WPAPSK=96389dc66eaf7e6efd5b5523ae43c7925ff4df2f8b70994
#WLAN_WPA[4]="wpa_supplicant"
#WLAN_WPADRIVER[4]="ndiswrapper"

## Some examples of additional network parameters that you can use.
## Config information for wlan0:
#IFNAME[4]="wlan0"              # Use a different interface name nstead of
                                # the default 'eth4'
#HWADDR[4]="00:01:23:45:67:89"  # Overrule the card's hardware MAC address
#MTU[4]=""                      # The default MTU is 1500, but you might need
                                # 1360 when you use NAT'ed IPSec traffic.
#DHCP_KEEPRESOLV[4]="yes"       # If you dont want /etc/resolv.conf overwritten
#DHCP_KEEPNTP[4]="yes"          # If you don't want ntp.conf overwritten
#DHCP_KEEPGW[4]="yes"           # If you don't want the DHCP server to change
                                # your default gateway
#DHCP_IPADDR[4]=""              # Request a specific IP address from the DHCP
                                # server
#WLAN_ESSID[4]=DARKSTAR         # Here, you can override _any_ parameter
                                # defined in rc.wireless.conf, by prepending
                                # 'WLAN_' to the parameter's name. Useful for
                                # those with multiple wireless interfaces.
#WLAN_IWPRIV[4]="set AuthMode=WPAPSK | set EncrypType=TKIP | set WPAPSK=thekey"
                                # Some drivers require a private ioctl to be
                                # set through the iwpriv command. If more than
                                # one is required, you can place them in the
                                # IWPRIV parameter (separated with the pipe (|)
                                # character, see the example).
```

23.2 /etc/dnsmasq.conf

Server

```
# Configuration file for dnsmasq.
#
# Format is one option per line, legal options are the same
# as the long options legal on the command line. See
# "/usr/sbin/dnsmasq --help" or "man 8 dnsmasq" for details.

# The following two options make you a better netizen, since they
# tell dnsmasq to filter out queries which the public DNS cannot
# answer, and which load the servers (especially the root servers)
# uneccessarily. If you have a dial-on-demand link they also stop
# these requests from bringing up the link uneccessarily.

# Never forward plain names (without a dot or domain part)
#domain-needed
# Never forward addresses in the non-routed address spaces.
#bogus-priv
```

```
# Uncomment this to filter useless windows-originated DNS requests
# which can trigger dial-on-demand links needlessly.
# Note that (amongst other things) this blocks all SRV requests,
# so don't use it if you use eg Kerberos, SIP, XMMP or Google-talk.
# This option only affects forwarding, SRV records originating for
# dnsmasq (via srv-host= lines) are not suppressed by it.
#filterwin2k

# Change this line if you want dns to get its upstream servers from
# somewhere other that /etc/resolv.conf
#resolv-file=

# By  default,  dnsmasq  will  send queries to any of the upstream
# servers it knows about and tries to favour servers to are  known
# to  be  up.  Uncommenting this forces dnsmasq to try each query
# with  each  server  strictly  in  the  order  they   appear   in
# /etc/resolv.conf
#strict-order

# If you don't want dnsmasq to read /etc/resolv.conf or any other
# file, getting its servers from this file instead (see below), then
# uncomment this.
#no-resolv

# If you don't want dnsmasq to poll /etc/resolv.conf or other resolv
# files for changes and re-read them then uncomment this.
#no-poll

# Add other name servers here, with domain specs if they are for
# non-public domains.
#server=/localnet/192.168.0.1

# Example of routing PTR queries to nameservers: this will send all
# address->name queries for 192.168.3/24 to nameserver 10.1.2.3
#server=/3.168.192.in-addr.arpa/10.1.2.3

# Add local-only domains here, queries in these domains are answered
# from /etc/hosts or DHCP only.
#local=/localnet/

# Add domains which you want to force to an IP address here.
# The example below send any host in doubleclick.net to a local
# webserver.
#address=/doubleclick.net/127.0.0.1

# --address (and --server) work with IPv6 addresses too.
#address=/www.thekelleys.org.uk/fe80::20d:60ff:fe36:f83

# You can control how dnsmasq talks to a server: this forces
# queries to 10.1.2.3 to be routed via eth1
# --server=10.1.2.3@eth1

# and this sets the source (ie local) address used to talk to
# 10.1.2.3 to 192.168.1.1 port 55 (there must be a interface with that
# IP on the machine, obviously).
# --server=10.1.2.3@192.168.1.1#55

# If you want dnsmasq to change uid and gid to something other
```

```
# than the default, edit the following lines.
#user=
#group=

# If you want dnsmasq to listen for DHCP and DNS requests only on
# specified interfaces (and the loopback) give the name of the
# interface (eg eth0) here.
# Repeat the line for more than one interface.
interface=eth1
# Or you can specify which interface _not_ to listen on
#except-interface=
# Or which to listen on by address (remember to include 127.0.0.1 if
# you use this.)
#listen-address=
# If you want dnsmasq to provide only DNS service on an interface,
# configure it as shown above, and then use the following line to
# disable DHCP on it.
#no-dhcp-interface=

# On systems which support it, dnsmasq binds the wildcard address,
# even when it is listening on only some interfaces. It then discards
# requests that it shouldn't reply to. This has the advantage of
# working even when interfaces come and go and change address. If you
# want dnsmasq to really bind only the interfaces it is listening on,
# uncomment this option. About the only time you may need this is when
# running another nameserver on the same machine.
#bind-interfaces

# If you don't want dnsmasq to read /etc/hosts, uncomment the
# following line.
#no-hosts
# or if you want it to read another file, as well as /etc/hosts, use
# this.
#addn-hosts=/etc/banner_add_hosts

# Set this (and domain: see below) if you want to have a domain
# automatically added to simple names in a hosts-file.
expand-hosts

# Set the domain for dnsmasq. this is optional, but if it is set, it
# does the following things.
# 1) Allows DHCP hosts to have fully qualified domain names, as long
#     as the domain part matches this setting.
# 2) Sets the "domain" DHCP option thereby potentially setting the
#    domain of all systems configured by DHCP
# 3) Provides the domain part for "expand-hosts"
#domain=thekelleys.org.uk
domain=ssf.net

# Uncomment this to enable the integrated DHCP server, you need
# to supply the range of addresses available for lease and optionally
# a lease time. If you have more than one network, you will need to
# repeat this for each network on which you want to supply DHCP
# service.
#dhcp-range=192.168.0.50,192.168.0.150,12h
dhcp-range=192.168.123.2,192.168.123.250,255.255.255.0,12h

# This is an example of a DHCP range where the netmask is given. This
# is needed for networks we reach the dnsmasq DHCP server via a relay
```

```
# agent. If you don't know what a DHCP relay agent is, you probably
# don't need to worry about this.
#dhcp-range=192.168.0.50,192.168.0.150,255.255.255.0,12h

# This is an example of a DHCP range with a network-id, so that
# some DHCP options may be set only for this network.
#dhcp-range=red,192.168.0.50,192.168.0.150

# Supply parameters for specified hosts using DHCP. There are lots
# of valid alternatives, so we will give examples of each. Note that
# IP addresses DO NOT have to be in the range given above, they just
# need to be on the same network. The order of the parameters in these
# do not matter, it's permissble to give name,adddress and MAC in any order

dhcp-host=*:*:*:*:*:*,ignore

# Always allocate the host with ethernet address 11:22:33:44:55:66
# The IP address 192.168.0.60
#dhcp-host=11:22:33:44:55:66,192.168.0.60
dhcp-host=00:11:11:82:EB:26,ssf32-02,192.168.123.2
dhcp-host=00:12:3F:A7:28:C3,ssf32-03,192.168.123.3
dhcp-host=00:07:E9:F0:C4:C9,ssf32-04,192.168.123.4
dhcp-host=00:14:22:3B:24:F3,ssf64-01,192.168.123.5
dhcp-host=00:16:36:EF:6E:8E,ssfGuest03,192.168.123.6
dhcp-host=00:17:F2:D4:59:3F,ssfGuest04,192.168.123.7
dhcp-host=00:16:36:2C:C5:23,ssfGuest05,192.168.123.8
dhcp-host=00:14:38:D4:48:CA,ssfNetPrinter01,192.168.123.9
dhcp-host=00:C0:9F:E8:0E:47,ssfGuest01,192.168.123.10
dhcp-host=00:16:D4:0E:54:F6,ssfGuest02,192.168.123.11
dhcp-host=00:14:22:8A:42:4A,laptopssf,192.168.123.12
dhcp-host=00:17:F2:D5:32:FE,ssfGuest06,192.168.123.13
dhcp-host=00:0F:1F:C4:22:14,ssfGuest07,192.168.123.14
dhcp-host=00:1B:24:17:FE:33,ssfGuest08,192.168.123.15
dhcp-host=00:16:36:E9:B7:28,ssfGuest09,192.168.123.16
dhcp-host=00:17:F2:31:3A:43,gavoxlaptop,192.168.123.17
dhcp-host=00:16:D4:16:59:92,ssfGuessPILAR,192.168.123.18
dhcp-host=00:1A:A0:E0:A8:85,SSFXeonQuad01,192.168.123.19
dhcp-host=00:1A:A0:E0:A6:B9,SSFXeonQuad02,192.168.123.20
dhcp-host=00:0F:1F:15:2E:17,ssfGuestHARISH,192.168.123.21
dhcp-host=00:1C:23:FF:23:6D,ssfGuestCosta,192.168.123.22
dhcp-host=00:1E:C9:03:CE:C6,ssfGuestfranco,192.168.123.23
dhcp-host=00:1D:09:4A:2C:A1,ssfGuestMiguel,192.168.123.24
dhcp-host=00:1F:3B:5A:B7:63,ssfGuestMiller,192.168.123.25
#dhcp-host=:::::,ssfGuestYeymy,192.168.123.26
#dhcp-host=00:16:D4:16:59:92,ssfGuessNelcy,192.168.123.19
#dhcp-host=:::::,,192.168.123.17

# Always set the name of the host with hardware address
# 11:22:33:44:55:66 to be "fred"
#dhcp-host=11:22:33:44:55:66,fred

# Always give the host with ethernet address 11:22:33:44:55:66
# the name fred and IP address 192.168.0.60 and lease time 45 minutes
#dhcp-host=11:22:33:44:55:66,fred,192.168.0.60,45m

# Give the machine which says its name is "bert" IP address
# 192.168.0.70 and an infinite lease
#dhcp-host=bert,192.168.0.70,infinite
```

```
# Always give the host with client identifier 01:02:02:04
# the IP address 192.168.0.60
#dhcp-host=id:01:02:02:04,192.168.0.60

# Always give the host with client identifier "marjorie"
# the IP address 192.168.0.60
#dhcp-host=id:marjorie,192.168.0.60

# Enable the address given for "judge" in /etc/hosts
# to be given to a machine presenting the name "judge" when
# it asks for a DHCP lease.
#dhcp-host=judge

# Never offer DHCP service to a machine whose ethernet
# address is 11:22:33:44:55:66
#dhcp-host=11:22:33:44:55:66,ignore

# Ignore any client-id presented by the machine with ethernet
# address 11:22:33:44:55:66. This is useful to prevent a machine
# being treated differently when running under different OS's or
# between PXE boot and OS boot.
#dhcp-host=11:22:33:44:55:66,id:*

# Send extra options which are tagged as "red" to
# the machine with ethernet address 11:22:33:44:55:66
#dhcp-host=11:22:33:44:55:66,net:red

# Send extra options which are tagged as "red" to
# any machine with ethernet address starting 11:22:33:
#dhcp-host=11:22:33:*:*:*,net:red

# Ignore any clients which are specified in dhcp-host lines
# or /etc/ethers. Equivalent to ISC "deny unkown-clients".
# This relies on the special "known" tag which is set when
# a host is matched.
#dhcp-ignore=#known

# Send extra options which are tagged as "red" to any machine whose
# DHCP vendorclass string includes the substring "Linux"
#dhcp-vendorclass=red,Linux

# Send extra options which are tagged as "red" to any machine one
# of whose DHCP userclass strings includes the substring "accounts"
#dhcp-userclass=red,accounts

# Send extra options which are tagged as "red" to any machine whose
# MAC address matches the pattern.
#dhcp-mac=red,00:60:8C:*:*:*

# If this line is uncommented, dnsmasq will read /etc/ethers and act
# on the ethernet-address/IP pairs found there just as if they had
# been given as --dhcp-host options. Useful if you keep
# MAC-address/host mappings there for other purposes.
#read-ethers

# Send options to hosts which ask for a DHCP lease.
# See RFC 2132 for details of available options.
# Common options can be given to dnsmasq by name:
# run "dnsmasq --help dhcp" to get a list.
```

```
# Note that all the common settings, such as netmask and
# broadcast address, DNS server and default route, are given
# sane defaults by dnsmasq. You very likely will not need
# any dhcp-options. If you use Windows clients and Samba, there
# are some options which are recommended, they are detailed at the
# end of this section.

dhcp-option=1,255.255.255.0
dhcp-option=6,192.168.123.1
dhcp-option=44,168.176.160.22,168.176.160.23
dhcp-option=41,192.168.123.1


# Override the default route supplied by dnsmasq, which assumes the
# router is the same machine as the one running dnsmasq.
#dhcp-option=3,1.2.3.4

# Do the same thing, but using the option name
#dhcp-option=option:router,1.2.3.4

# Override the default route supplied by dnsmasq and send no default
# route at all. Note that this only works for the options sent by
# default (1, 3, 6, 12, 28) the same line will send a zero-length option
# for all other option numbers.
#dhcp-option=3

# Set the NTP time server addresses to 192.168.0.4 and 10.10.0.5
#dhcp-option=option:ntp-server,192.168.0.4,10.10.0.5
dhcp-option=option:ntp-server,192.168.123.1

# Set the NTP time server address to be the same machine as
# is running dnsmasq
dhcp-option=42,0.0.0.0

# Set the NIS domain name to "welly"
#dhcp-option=40,welly
dhcp-option=40,ssfservernis

# Set the default time-to-live to 50
#dhcp-option=23,50

# Set the "all subnets are local" flag
#dhcp-option=27,1

# Send the etherboot magic flag and then etherboot options (a string).
#dhcp-option=128,e4:45:74:68:00:00
#dhcp-option=129,NIC=eepro100

# Specify an option which will only be sent to the "red" network
# (see dhcp-range for the declaration of the "red" network)
# Note that the net: part must precede the option: part.
#dhcp-option = net:red, option:ntp-server, 192.168.1.1

# The following DHCP options set up dnsmasq in the same way as is specified
# for the ISC dhcpcd in
# http://www.samba.org/samba/ftp/docs/textdocs/DHCP-Server-Configuration.txt
# adapted for a typical dnsmasq installation where the host running
# dnsmasq is also the host running samba.
# you may want to uncomment them if you use Windows clients and Samba.
```

```
#dhcp-option=19,0           # option ip-forwarding off
#dhcp-option=44,0.0.0.0     # set netbios-over-TCP/IP nameserver(s) aka WINS server(s)
#dhcp-option=45,0.0.0.0     # netbios datagram distribution server
#dhcp-option=46,8           # netbios node type
#dhcp-option=47             # empty netbios scope.

# Send RFC-3397 DNS domain search DHCP option. WARNING: Your DHCP client
# probably doesn't support this......
#dhcp-option=option:domain-search,eng.apple.com,marketing.apple.com

# Send RFC-3442 classless static routes (note the netmask encoding)
#dhcp-option=121,192.168.1.0/24,1.2.3.4,10.0.0.0/8,5.6.7.8

# Send vendor-class specific options encapsulated in DHCP option 43.
# The meaning of the options is defined by the vendor-class so
# options are sent only when the client supplied vendor class
# matches the class given here. (A substring match is OK, so "MSFT"
# matches "MSFT" and "MSFT 5.0"). This example sets the
# mtftp address to 0.0.0.0 for PXEClients.
#dhcp-option=vendor:PXEClient,1,0.0.0.0

# Send microsoft-specific option to tell windows to release the DHCP lease
# when it shuts down. Note the "i" flag, to tell dnsmasq to send the
# value as a four-byte integer - that's what microsoft wants. See
# http://technet2.microsoft.com/WindowsServer/en/library/a70f1bb7-d2d4-49f0-96d6-4b7414ecfaae1033.mspx?mfr=true
#dhcp-option=vendor:MSFT,2,1i

# Send the Encapsulated-vendor-class ID needed by some configurations of
# Etherboot to allow is to recognise the DHCP server.
#dhcp-option=vendor:Etherboot,60,"Etherboot"

# Send options to PXELinux. Note that we need to send the options even
# though they don't appear in the parameter request list, so we need
# to use dhcp-option-force here.
# See http://syslinux.zytor.com/pxe.php#special for details.
# Magic number - needed before anything else is recognised
#dhcp-option-force=208,f1:00:74:7e
# Configuration file name
#dhcp-option-force=209,configs/common
# Path prefix
#dhcp-option-force=210,/tftpboot/pxelinux/files/
# Reboot time. (Note 'i' to send 32-bit value)
#dhcp-option-force=211,30i

# Set the boot filename for BOOTP. You will only need
# this is you want to boot machines over the network and you will need
# a TFTP server; either dnsmasq's built in TFTP server or an
# external one. (See below for how to enable the TFTP server.)
#dhcp-boot=pxelinux.0

# Boot for Etherboot gPXE. The idea is to send two different
# filenames, the first loads gPXE, and the second tells gPXE what to
# load. The dhcp-match sets the gpxe tag for requests from gPXE.
#dhcp-match=gpxe,175 # gPXE sends a 175 option.
#dhcp-boot=net:#gpxe,undionly.kpxe
#dhcp-boot=mybootimage

# Enable dnsmasq's built-in TFTP server
#enable-tftp
```

```
# Set the root directory for files availble via FTP.
#tftp-root=/var/ftpd

# Make the TFTP server more secure: with this set, only files owned by
# the user dnsmasq is running as will be send over the net.
#tftp-secure

# Set the boot file name only when the "red" tag is set.
#dhcp-boot=net:red,pxelinux.red-net

# An example of dhcp-boot with an external server: the name and IP
# address of the server are given after the filename.
#dhcp-boot=/var/ftpd/pxelinux.0,boothost,192.168.0.3

# Set the limit on DHCP leases, the default is 150
#dhcp-lease-max=150

# The DHCP server needs somewhere on disk to keep its lease database.
# This defaults to a sane location, but if you want to change it, use
# the line below.
#dhcp-leasefile=/var/state/dnsmasq/dnsmasq.leases

# Set the DHCP server to authoritative mode. In this mode it will barge in
# and take over the lease for any client which broadcasts on the network,
# whether it has a record of the lease or not. This avoids long timeouts
# when a machine wakes up on a new network. DO NOT enable this if there's
# the slighest chance that you might end up accidentally configuring a DHCP
# server for your campus/company accidentally. The ISC server uses
# the same option, and this URL provides more information:
# http://www.isc.org/index.pl?/sw/dhcp/authoritative.php
#dhcp-authoritative

# Run an executable when a DHCP lease is created or destroyed.
# The arguments sent to the script are "add" or "del",
# then the MAC address, the IP address and finally the hostname
# if there is one.
#dhcp-script=/bin/echo

# Set the cachesize here.
#cache-size=150

# If you want to disable negative caching, uncomment this.
#no-negcache

# Normally responses which come form /etc/hosts and the DHCP lease
# file have Time-To-Live set as zero, which conventionally means
# do not cache further. If you are happy to trade lower load on the
# server for potentially stale date, you can set a time-to-live (in
# seconds) here.
#local-ttl=

# If you want dnsmasq to detect attempts by Verisign to send queries
# to unregistered .com and .net hosts to its sitefinder service and
# have dnsmasq instead return the correct NXDOMAIN response, uncomment
# this line. You can add similar lines to do the same for other
# registries which have implemented wildcard A records.
#bogus-nxdomain=64.94.110.11
```

```
# If you want to fix up DNS results from upstream servers, use the
# alias option. This only works for IPv4.
# This alias makes a result of 1.2.3.4 appear as 5.6.7.8
#alias=1.2.3.4,5.6.7.8
# and this maps 1.2.3.x to 5.6.7.x
#alias=1.2.3.0,5.6.7.0,255.255.255.0


# Change these lines if you want dnsmasq to serve MX records.

# Return an MX record named "maildomain.com" with target
# servermachine.com and preference 50
#mx-host=maildomain.com,servermachine.com,50

# Set the default target for MX records created using the localmx option.
#mx-target=servermachine.com

# Return an MX record pointing to the mx-target for all local
# machines.
#localmx

# Return an MX record pointing to itself for all local machines.
#selfmx

# Change the following lines if you want dnsmasq to serve SRV
# records.  These are useful if you want to serve ldap requests for
# Active Directory and other windows-originated DNS requests.
# See RFC 2782.
# You may add multiple srv-host lines.
# The fields are <name>,<target>,<port>,<priority>,<weight>
# If the domain part if missing from the name (so that is just has the
# service and protocol sections) then the domain given by the domain=
# config option is used. (Note that expand-hosts does not need to be
# set for this to work.)

# A SRV record sending LDAP for the example.com domain to
# ldapserver.example.com port 289
#srv-host=_ldap._tcp.example.com,ldapserver.example.com,389

# A SRV record sending LDAP for the example.com domain to
# ldapserver.example.com port 289 (using domain=)
#domain=example.com
#srv-host=_ldap._tcp,ldapserver.example.com,389

# Two SRV records for LDAP, each with different priorities
#srv-host=_ldap._tcp.example.com,ldapserver.example.com,389,1
#srv-host=_ldap._tcp.example.com,ldapserver.example.com,389,2

# A SRV record indicating that there is no LDAP server for the domain
# example.com
#srv-host=_ldap._tcp.example.com

# The following line shows how to make dnsmasq serve an arbitrary PTR
# record. This is useful for DNS-SD. (Note that the
# domain-name expansion done for SRV records _does_not
# occur for PTR records.)
#ptr-record=_http._tcp.dns-sd-services,"New Employee Page._http._tcp.dns-sd-services"

# Change the following lines to enable dnsmasq to serve TXT records.
```

```
# These are used for things like SPF and zeroconf. (Note that the
# domain-name expansion done for SRV records _does_not
# occur for TXT records.)

#Example SPF.
#txt-record=example.com,"v=spf1 a -all"

#Example zeroconf
#txt-record=_http._tcp.example.com,name=value,paper=A4


# For debugging purposes, log each DNS query as it passes through
# dnsmasq.
log-queries

# Log lots of extra information about DHCP transactions.
log-dhcp

# Include a another lot of configuration options.
#conf-file=/etc/dnsmasq.more.conf
#conf-dir=/etc/dnsmasq.d
```

23.3 /etc/exports

Server

```
# See exports(5) for a description.
# This file contains a list of all directories exported to other computers.
# It is used by rpc.nfsd and rpc.mountd.

/home 192.168.123.0/255.255.255.0(rw,sync,no_root_squash)
```

23.4 /etc/hosts.deny

Server

```
#
# hosts.deny This file describes the names of the hosts which are
# *not* allowed to use the local INET services, as decided
# by the '/usr/sbin/tcpd' server.
#
# Version: @(#)/etc/hosts.deny 1.00 05/28/93
#
# Author: Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org
#
#

# End of hosts.deny.
```

23.5 /etc/hosts.allow

Server

```
#
# hosts.allow This file describes the names of the hosts which are
# allowed to use the local INET services, as decided by
# the '/usr/sbin/tcpd' server.
#
# Version: @(#)/etc/hosts.allow 1.00 05/28/93
#
# Author: Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org
#
#
portmap:ALL
lockd:ALL
mountd:ALL
rquotad:ALL
statd:ALL
sshd:ALL
#sshd: 168.176.14.52
# End of hosts.allow.
```

23.6 /etc/fstab

Client

```
/dev/sda3        swap             swap        defaults         0   0
/dev/sda2        /                reiserfs    defaults         1   1
/dev/sda5        /mnt/local/data01 reiserfs    defaults         1   2
/dev/sda6        /mnt/local/data02 reiserfs    defaults         1   2
#/dev/cdrom      /mnt/cdrom       auto        noauto,owner,ro  0   0
/dev/fd0         /mnt/floppy      auto        noauto,owner     0   0
devpts           /dev/pts         devpts      gid=5,mode=620   0   0
proc             /proc            proc        defaults         0   0
tmpfs            /dev/shm         tmpfs       defaults         0   0
#NFS VOLUMES
192.168.123.1:/home /home nfs rw,hard,intr 0 0
```

23.7 /etc/defaultdomain

Server

```
ssfservernis
```

23.8 /var/yp/Makefile

Server

```
#
# Makefile for the NIS databases
#
# This Makefile should only be run on the NIS master server of a domain.
# All updated maps will be pushed to all NIS slave servers listed in the
# /var/yp/ypservers file. Please make sure that the hostnames of all
# NIS servers in your domain are listed in /var/yp/ypservers.
#
# This Makefile can be modified to support more NIS maps if desired.
#

# Set the following variable to "-b" to have NIS servers use the domain
# name resolver for hosts not in the current domain. This is only needed,
# if you have SunOS slave YP server, which gets here maps from this
# server. The NYS YP server will ignore the YP_INTERDOMAIN key.
#B=-b
B=

# If we have only one server, we don't have to push the maps to the
# slave servers (NOPUSH=true). If you have slave servers, change this
# to "NOPUSH=false" and put all hostnames of your slave servers in the file
# /var/yp/ypservers.
NOPUSH=true

# We do not put password entries with lower UIDs (the root and system
# entries) in the NIS password database, for security. MINUID is the
# lowest uid that will be included in the password maps. If you
# create shadow maps, the UserID for a shadow entry is taken from
# the passwd file. If no entry is found, this shadow entry is
# ignored.
# MINGID is the lowest gid that will be included in the group maps.
MINUID=500
#MINGID=500
MINGID=1

# Should we merge the passwd file with the shadow file ?
# MERGE_PASSWD=true|false
MERGE_PASSWD=false

# Should we merge the group file with the gshadow file ?
# MERGE_GROUP=true|false
MERGE_GROUP=false

# These are commands which this Makefile needs to properly rebuild the
# NIS databases. Don't change these unless you have a good reason.
AWK = /usr/bin/gawk
MAKE = /usr/bin/gmake
UMASK = umask 066

#
# These are the source directories for the NIS files; normally
# that is /etc but you may want to move the source for the password
# and group files to (for example) /var/yp/ypfiles. The directory
# for passwd, group and shadow is defined by YPPWDDIR, the rest is
# taken from YPSRCDIR.
#
YPSRCDIR = /etc
YPPWDDIR = /etc
YPBINDIR = /usr/lib/yp
```

```
YPSBINDIR = /usr/sbin
YPDIR = /var/yp
YPMAPDIR = $(YPDIR)/$(DOMAIN)

# These are the files from which the NIS databases are built. You may edit
# these to taste in the event that you wish to keep your NIS source files
# seperate from your NIS server's actual configuration files.
#
GROUP       = $(YPPWDDIR)/group
PASSWD      = $(YPPWDDIR)/passwd
SHADOW     = $(YPPWDDIR)/shadow
GSHADOW     = $(YPPWDDIR)/gshadow
ADJUNCT     = $(YPPWDDIR)/passwd.adjunct
#ALIASES     = $(YPSRCDIR)/aliases  # aliases could be in /etc or /etc/mail
ALIASES     = /etc/mail/aliases
ETHERS      = $(YPSRCDIR)/ethers     # ethernet addresses (for rarpd)
BOOTPARAMS  = $(YPSRCDIR)/bootparams # for booting Sun boxes (bootparamd)
HOSTS       = $(YPSRCDIR)/hosts
NETWORKS    = $(YPSRCDIR)/networks
PRINTCAP    = $(YPSRCDIR)/printcap
PROTOCOLS   = $(YPSRCDIR)/protocols
PUBLICKEYS  = $(YPSRCDIR)/publickey
RPC      = $(YPSRCDIR)/rpc
SERVICES    = $(YPSRCDIR)/services
NETGROUP    = $(YPSRCDIR)/netgroup
NETID     = $(YPSRCDIR)/netid
AMD_HOME    = $(YPSRCDIR)/amd.home
AUTO_MASTER = $(YPSRCDIR)/auto.master
AUTO_HOME   = $(YPSRCDIR)/auto.home
AUTO_LOCAL  = $(YPSRCDIR)/auto.local
TIMEZONE    = $(YPSRCDIR)/timezone
LOCALE      = $(YPSRCDIR)/locale
NETMASKS    = $(YPSRCDIR)/netmasks

YPSERVERS = $(YPDIR)/ypservers # List of all NIS servers for a domain

target: Makefile
@test ! -d $(LOCALDOMAIN) && mkdir $(LOCALDOMAIN) ; \
cd $(LOCALDOMAIN)  ; \
$(NOPUSH) || $(MAKE) -f ../Makefile ypservers; \
$(MAKE) -f ../Makefile all

# If you don't want some of these maps built, feel free to comment
# them out from this list.

all:  passwd group hosts rpc services netid protocols netgrp mail \
shadow #publickey # networks ethers bootparams printcap \
# amd.home auto.master auto.home auto.local passwd.adjunct \
# timezone locale netmasks


############################################################################
#                                                                          #
#  DON'T EDIT ANYTHING BELOW IF YOU DON'T KNOW WHAT YOU ARE DOING !!!  #
#                                                                          #
############################################################################

DBLOAD = $(YPBINDIR)/makedbm -c -m `$(YPBINDIR)/yphelper --hostname`
MKNETID = $(YPBINDIR)/mknetid
```

```
YPPUSH = $(YPSBINDIR)/yppush
MERGER = $(YPBINDIR)/yphelper
DOMAIN = `basename \`pwd\``
LOCALDOMAIN = `/bin/domainname`
REVNETGROUP = $(YPBINDIR)/revnetgroup
CREATE_PRINTCAP = $(YPBINDIR)/create_printcap

ethers:     ethers.byname ethers.byaddr
hosts:     hosts.byname hosts.byaddr
networks:   networks.byaddr networks.byname
protocols:  protocols.bynumber protocols.byname
rpc:     rpc.byname rpc.bynumber
services:   services.byname services.byservicename
passwd:     passwd.byname passwd.byuid
group:      group.byname group.bygid
shadow:     shadow.byname
passwd.adjunct: passwd.adjunct.byname
netid:     netid.byname
netgrp:     netgroup netgroup.byhost netgroup.byuser
publickey:  publickey.byname
mail:     mail.aliases
timezone:      timezone.byname
locale:                locale.byname
netmasks:      netmasks.byaddr

ypservers: $(YPSERVERS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#") print $$0"\t"$$0 }' \
    $(YPSERVERS) | $(DBLOAD) -i $(YPSERVERS) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

$(YPSERVERS):
@echo -n "Generating $*..."
@uname -n > $(YPSERVERS)

bootparams: $(BOOTPARAMS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#" && $$1 != "+") \
print $$0 }' $(BOOTPARAMS) | $(DBLOAD) -r -i $(BOOTPARAMS) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


ethers.byname: $(ETHERS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#" && $$1 != "+") \
print $$2"\t"$$0 }' $(ETHERS) | $(DBLOAD) -r -i $(ETHERS) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


ethers.byaddr: $(ETHERS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#" && $$1 != "+") \
print $$1"\t"$$0 }' $(ETHERS) | $(DBLOAD) -r -i $(ETHERS) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
```

```
netgroup: $(NETGROUP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#" && $$1 != "+") \
print $$0 }' $(NETGROUP) | $(DBLOAD) -i $(NETGROUP) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


netgroup.byhost: $(NETGROUP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(REVNETGROUP) -h < $(NETGROUP) | $(DBLOAD) -i $(NETGROUP) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


netgroup.byuser: $(NETGROUP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(REVNETGROUP) -u < $(NETGROUP) | $(DBLOAD) -i $(NETGROUP) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


hosts.byname: $(HOSTS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '/^[0-9]/ { for (n=2; n<=NF && $$n !~ "#"; n++) \
print $$n"\t"$$0 }' $(HOSTS) | $(DBLOAD) -r $(B) -l \
-i $(HOSTS) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

hosts.byaddr: $(HOSTS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") print $$1"\t"$$0 }' \
   $(HOSTS) | $(DBLOAD) -r $(B) -i $(HOSTS) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


networks.byname: $(NETWORKS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if($$1 !~ "#" && $$1 != "") { print $$1"\t"$$0; \
 for (n=3; n<=NF && $$n !~ "#"; n++) print $$n"\t"$$0 \
}}' $(NETWORKS) | $(DBLOAD) -r -i $(NETWORKS) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


networks.byaddr: $(NETWORKS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") print $$2"\t"$$0 }' \
 $(NETWORKS) | $(DBLOAD) -r -i $(NETWORKS) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


protocols.byname: $(PROTOCOLS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") { print $$1"\t"$$0; \
for (n=3; n<=NF && $$n !~ "#"; n++) \
print $$n"\t"$$0}}' $(PROTOCOLS) | $(DBLOAD) -r -i \
$(PROTOCOLS) -o $(YPMAPDIR)/$@ - $@
```

```
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


protocols.bynumber: $(PROTOCOLS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") print $$2"\t"$$0 }' \
$(PROTOCOLS) | $(DBLOAD) -r -i $(PROTOCOLS) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


rpc.byname: $(RPC) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#"  && $$1 != "") { print $$1"\t"$$0; \
for (n=3; n<=NF && $$n !~ "#"; n++)  print $$n"\t"$$0 \
  }}' $(RPC) | $(DBLOAD) -r -i $(RPC) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


rpc.bynumber: $(RPC) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") print $$2"\t"$$0 }' $(RPC) \
| $(DBLOAD) -r -i $(RPC) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


services.byname: $(SERVICES) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") print $$2"\t"$$0 }' \
$(SERVICES) | $(DBLOAD) -r -i $(SERVICES) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

services.byservicename: $(SERVICES) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 !~ "#" && $$1 != "") { \
split($$2,A,"/") ; TMP = "/" A[2] ; \
print $$1 TMP"\t"$$0 ; \
if (! seen[$$1]) { seen[$$1] = 1 ; print $$1"\t"$$0 ; } \
for (N = 3; N <= NF && $$N !~ "#" ; N++) { \
if ($$N !~ "#" && $$N != "") print $$N TMP"\t"$$0 ; \
if (! seen[$$N]) { seen[$$N] = 1 ; print $$N"\t"$$0 ; } \
} } } ' \
$(SERVICES) | $(DBLOAD) -r -i $(SERVICES) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


ifeq (x$(MERGE_PASSWD),xtrue)
passwd.byname: $(PASSWD) $(SHADOW) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(MERGER) -p $(PASSWD) $(SHADOW) | \
   $(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINUID) ) \
   print $$1"\t"$$0 }' | $(DBLOAD) -i $(PASSWD) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

passwd.byuid: $(PASSWD) $(SHADOW) $(YPDIR)/Makefile
```

```
@echo "Updating $@..."
@$(UMASK); \
$(MERGER) -p $(PASSWD) $(SHADOW) | \
   $(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINUID) ) \
   print $$3"\t"$$0 }' | $(DBLOAD) -i $(PASSWD) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

# Don't build a shadow map !
shadow.byname:
@echo "Updating $@... Ignored -> merged with passwd"

else

passwd.byname: $(PASSWD) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINUID) ) \
   print $$1"\t"$$0 }' $(PASSWD) | $(DBLOAD) -i $(PASSWD) \
-o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

passwd.byuid: $(PASSWD) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINUID) ) \
   print $$3"\t"$$0 }' $(PASSWD) | $(DBLOAD) -i $(PASSWD) \
 -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

shadow.byname: $(SHADOW) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '{ if (FILENAME ~ /shadow$$/) { \
if (UID[$$1] >= $(MINUID) ) print $$1"\t"$$0; \
} else UID[$$1] = $$3; }' $(PASSWD) $(SHADOW) \
| $(DBLOAD) -s -i $(SHADOW) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
endif

passwd.adjunct.byname: $(ADJUNCT) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" ) print $$1"\t"$$0 }' \
$(ADJUNCT) | $(DBLOAD) -s -i $(ADJUNCT) -o $(YPMAPDIR)/$@ - $@
@chmod 700 $(YPDIR)/$(DOMAIN)/$@*
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

ifeq (x$(MERGE_GROUP),xtrue)
group.byname: $(GROUP) $(GSHADOW) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(MERGER) -g $(GROUP) $(GSHADOW) | \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINGID) ) \
print $$1"\t"$$0 }' | $(DBLOAD) -i $(GROUP) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

group.bygid: $(GROUP) $(GSHADOW) $(YPDIR)/Makefile
@echo "Updating $@..."
```

```
@$(UMASK); \
$(MERGER) -g $(GROUP) $(GSHADOW) | \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINGID) ) \
print $$3"\t"$$0 }' | $(DBLOAD) -i $(GROUP) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

else

group.byname: $(GROUP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINGID) ) \
print $$1"\t"$$0 }' $(GROUP) \
| $(DBLOAD) -i $(GROUP) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

group.bygid: $(GROUP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(UMASK); \
$(AWK) -F: '!/^[-+#]/ { if ($$1 != "" && $$3 >= $(MINGID) ) \
print $$3"\t"$$0 }' $(GROUP) \
| $(DBLOAD) -i $(GROUP) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
endif

$(NETID):
netid.byname: $(GROUP) $(PASSWD) $(HOSTS) $(NETID) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(MKNETID) -q -p $(PASSWD) -g $(GROUP) -h $(HOSTS) -d $(DOMAIN) \
-n $(NETID) | $(DBLOAD) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


mail.aliases: $(ALIASES) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ \
if ($$1 ~ "^#.*") \
next; \
if ($$1 == "" || $$1 == "+") { \
if (line != "") \
{print line; line = "";} \
next; \
} \
if ($$0 ~ /^[[:space:]]/) \
line = line $$0; \
else { \
if (line != "") \
{print line; line = "";} \
line = $$0; \
} \
} \
END {if (line != "") print line}' \
$(ALIASES) | $(DBLOAD) --aliases \
-i $(ALIASES) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


publickey.byname: $(PUBLICKEYS) $(YPDIR)/Makefile
@echo "Updating $@..."
```

```
@$(AWK) '{ if($$1 !~ "#" && $$1 != "") { print $$1"\t"$$2 }}' \
$(PUBLICKEYS) | $(DBLOAD) -i $(PUBLICKEYS) \
 -o $(YPMAPDIR)/$@ - $@
@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


printcap: $(PRINTCAP) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(CREATE_PRINTCAP) < $(PRINTCAP) | \
$(DBLOAD) -i $(PRINTCAP) -o $(YPMAPDIR)/$@ - $@
@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


auto.master: $(AUTO_MASTER) $(YPDIR)/Makefile
@echo "Updating $@..."
-@sed -e "/^#/d" -e s/#.*$$// $(AUTO_MASTER) | $(DBLOAD) \
-i $(AUTO_MASTER) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

auto.home: $(AUTO_HOME) $(YPDIR)/Makefile
@echo "Updating $@..."
-@sed -e "/^#/d" -e s/#.*$$// $(AUTO_HOME) | $(DBLOAD) \
-i $(AUTO_HOME) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


auto.local: $(AUTO_LOCAL) $(YPDIR)/Makefile
@echo "Updating $@..."
-@sed -e "/^#/d" -e s/#.*$$// $(AUTO_LOCAL) | $(DBLOAD) \
-i $(AUTO_LOCAL) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


amd.home: $(AMD_HOME) $(YPDIR)/Makefile
@echo "Updating $@..."
-@sed -e "s/#.*$$//" -e "/^$$/d" $(AMD_HOME) | \
$(AWK) '{\
for (i = 1; i <= NF; i++)\
   if (i == NF) { \
      if (substr($$i, length($$i), 1) == "\\") \
                  printf("%s", substr($$i, 1, length($$i) -1)); \
               else \
  printf("%s\n",$$i); \
             } \
   else \
      printf("%s ",$$i);\
}' | $(DBLOAD) -i $(AMD_HOME) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

timezone.byname: $(TIMEZONE) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#") \
print $$2"\t"$$0 }' $(TIMEZONE) | $(DBLOAD) \
-r -i $(TIMEZONE) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


locale.byname: $(LOCALE) $(YPDIR)/Makefile
@echo "Updating $@..."
```

```
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#") \
     print $$2"\t"$$0"\n"$$1"\t"$$2"\t"$$1 }' $(LOCALE) | $(DBLOAD) \
-r -i $(LOCALE) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@


netmasks.byaddr: $(NETMASKS) $(YPDIR)/Makefile
@echo "Updating $@..."
@$(AWK) '{ if ($$1 != "" && $$1 !~ "#") \
print $$1"\t"$$2 }' $(NETMASKS) | $(DBLOAD) \
-r -i $(NETMASKS) -o $(YPMAPDIR)/$@ - $@
-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
```

23.9 /etc/rc.d/rc.yp

Server

```
#!/bin/sh
# /etc/rc.d/rc.yp
#
# Start NIS (Network Information Service).  NIS provides network-wide
# distribution of hostname, username, and other information databases.
# After configuring NIS, you will need to uncomment the parts of this
# script that you want to run.
#
# NOTE: for detailed information about setting up NIS, see the
# documentation in /usr/doc/yp-tools, /usr/doc/ypbind,
# /usr/doc/ypserv, and /usr/doc/Linux-HOWTOs/NIS-HOWTO.

# # First, we must setup the NIS domainname.  NOTE: this is not necessarily
# # the same as your DNS domainname, set in /etc/resolv.conf.  The NIS
# # domainname is the name of a domain served by your NIS server.
#
 if [ -r /etc/defaultdomain ]; then
   nisdomainname `cat /etc/defaultdomain`
 fi
#
# # NIS SERVER CONFIGURATION:
# # If you are the master server for the NIS domain, you must run ypserv to
# # service clients on the domain.
 if [ -x /usr/sbin/ypserv ]; then
   echo "Starting NIS server:  /usr/sbin/ypserv"
   /usr/sbin/ypserv
 fi
#
# # If you are the master server for the NIS domain, you must also run
# # rpc.yppasswdd, which is the RPC server that lets users change their
# # passwords.  You might also want users to be able to change their shell
# # and GECOS information, in which case you should comment out the first
# # yppasswdd line and uncomment out the second one.
 if [ -x /usr/sbin/rpc.yppasswdd ]; then
   echo "Starting NIS master password server:  /usr/sbin/rpc.yppasswdd"
   /usr/sbin/rpc.yppasswdd
#   # echo "Starting NIS master password server:  /usr/sbin/rpc.yppasswdd -e chsh -e chfn"
#   # /usr/sbin/rpc.yppasswdd -e chsh -e chfn
 fi
```

```
#
# # If you have NIS slave servers, you might also want to start up
# # rpc.ypxfrd, which transfers changes in the NIS domain to slave servers.
# # Alternatively, rpc.ypxfrd can be run out of inetd.
# if [ -x /usr/sbin/rpc.ypxfrd ]; then
#   echo "Starting NIS transfer server:  /usr/sbin/rpc.ypxfrd"
#   /usr/sbin/rpc.ypxfrd
# fi
#
# # NIS CLIENT CONFIGURATION:
# # If you are a NIS client, all you need to do is run ypbind, which will
# # broadcast across the network to find a server.  Your NIS server might
# # also be a client.
# if [ -d /var/yp ]; then
#   echo "Starting NIS services:  /usr/sbin/ypbind -broadcast"
#   /usr/sbin/ypbind -broadcast
# fi
#
# # Done setting up NIS.
```

23.10 /etc/rc.d/rc.yp

Client

```
#!/bin/sh
# /etc/rc.d/rc.yp
#
# Start NIS (Network Information Service).  NIS provides network-wide
# distribution of hostname, username, and other information databases.
# After configuring NIS, you will need to uncomment the parts of this
# script that you want to run.
#
# NOTE: for detailed information about setting up NIS, see the
# documentation in /usr/doc/yp-tools, /usr/doc/ypbind,
# /usr/doc/ypserv, and /usr/doc/Linux-HOWTOs/NIS-HOWTO.

# # First, we must setup the NIS domainname.  NOTE: this is not necessarily
# # the same as your DNS domainname, set in /etc/resolv.conf.  The NIS
# # domainname is the name of a domain served by your NIS server.
#
 if [ -r /etc/defaultdomain ]; then
   nisdomainname `cat /etc/defaultdomain`
 fi
#
# # NIS SERVER CONFIGURATION:
# # If you are the master server for the NIS domain, you must run ypserv to
# # service clients on the domain.
# if [ -x /usr/sbin/ypserv ]; then
#   echo "Starting NIS server:  /usr/sbin/ypserv"
#   /usr/sbin/ypserv
# fi
#
# # If you are the master server for the NIS domain, you must also run
# # rpc.yppasswdd, which is the RPC server that lets users change their
# # passwords.  You might also want users to be able to change their shell
# # and GECOS information, in which case you should comment out the first
```

```
# # yppasswdd line and uncomment out the second one.
# if [ -x /usr/sbin/rpc.yppasswdd ]; then
#   echo "Starting NIS master password server:  /usr/sbin/rpc.yppasswdd"
#   /usr/sbin/rpc.yppasswdd
#   # echo "Starting NIS master password server:  /usr/sbin/rpc.yppasswdd -e chsh -e chfn"
#   # /usr/sbin/rpc.yppasswdd -e chsh -e chfn
# fi
#
# # If you have NIS slave servers, you might also want to start up
# # rpc.ypxfrd, which transfers changes in the NIS domain to slave servers.
# # Alternatively, rpc.ypxfrd can be run out of inetd.
# if [ -x /usr/sbin/rpc.ypxfrd ]; then
#   echo "Starting NIS transfer server:  /usr/sbin/rpc.ypxfrd"
#   /usr/sbin/rpc.ypxfrd
# fi
#
# # NIS CLIENT CONFIGURATION:
# # If you are a NIS client, all you need to do is run ypbind, which will
# # broadcast across the network to find a server.  Your NIS server might
# # also be a client.
 if [ -d /var/yp ]; then
   echo "Starting NIS services:  /usr/sbin/ypbind -broadcast"
   /usr/sbin/ypbind -broadcast
 fi
#
# # Done setting up NIS.
```

23.11 /etc/nsswitch.conf

Client

```
#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reason
# (like no NIS server responding) then the search continues with the
# next entry.
#
# Legal entries are:
#
#      nisplus or nis+         Use NIS+ (NIS version 3)
#      nis or yp               Use NIS (NIS version 2), also called YP
#      dns                     Use DNS (Domain Name Service)
#      files                   Use the local files
#      [NOTFOUND=return]       Stop searching if not found so far
#

 passwd:  files nis
 shadow:  files nis
# group:   files nis
 group:   nis files
```

```
#passwd:  compat
#group:   compat

hosts:   files dns
networks:       files

services:       files
protocols:      files
rpc:     files
ethers:  files
netmasks:       files
netgroup:       files
bootparams:     files

automount:      files
aliases:        files
```

23.12 /etc/passwd

Client

```
root:x:0:0::/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/log:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/:/bin/false
news:x:9:13:news:/usr/lib/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucppublic:/bin/false
operator:x:11:0:operator:/root:/bin/bash
games:x:12:100:games:/usr/games:/bin/false
ftp:x:14:50::/home/ftp:/bin/false
smmsp:x:25:25:smmsp:/var/spool/clientmqueue:/bin/false
mysql:x:27:27:MySQL:/var/lib/mysql:/bin/false
rpc:x:32:32:RPC portmap user:/:/bin/false
sshd:x:33:33:sshd:/:/bin/false
gdm:x:42:42:GDM:/var/state/gdm:/bin/bash
apache:x:80:80:User for Apache:/srv/httpd:/bin/false
messagebus:x:81:81:User for D-BUS:/var/run/dbus:/bin/false
haldaemon:x:82:82:User for HAL:/var/run/hald:/bin/false
pop:x:90:90:POP:/:/bin/false
nobody:x:99:99:nobody:/:/bin/false
+::::::
```

23.13 /etc/shadow

Client

```
root:$1$p6215XMc$yaRIqoDl2z25YkPNOSMXQ.:14103:0:::::
bin:*:9797:0:::::
daemon:*:9797:0:::::
adm:*:9797:0:::::
lp:*:9797:0:::::
sync:*:9797:0:::::
shutdown:*:9797:0:::::
halt:*:9797:0:::::
mail:*:9797:0:::::
news:*:9797:0:::::
uucp:*:9797:0:::::
operator:*:9797:0:::::
games:*:9797:0:::::
ftp:*:9797:0:::::
smmsp:*:9797:0:::::
mysql:*:9797:0:::::
rpc:*:9797:0:::::
sshd:*:9797:0:::::
gdm:*:9797:0:::::
pop:*:9797:0:::::
apache:*:9797:0:::::
messagebus:*:9797:0:::::
haldaemon:*:9797:0:::::
nobody:*:9797:0:::::
+::::::::
```

23.14 /etc/group

Client

```
root:x:0:root
bin:x:1:root,bin
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root,adm
lp:x:7:lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
floppy:x:11:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
audio:x:17:root
video:x:18:root
cdrom:x:19:root
games:x:20:
slocate:x:21:
utmp:x:22:
smmsp:x:25:smmsp
tape:x:26:root
mysql:x:27:
```

```
rpc:x:32:
sshd:x:33:sshd
gdm:x:42:
shadow:x:43:
ftp:x:50:
apache:x:80:
messagebus:x:81:
haldaemon:x:82:
plugdev:x:83:root
power:x:84:
pop:x:90:pop
scanner:x:93:
nobody:x:98:nobody
nogroup:x:99:
users:x:100:
console:x:101:
+:::
```

23.15 /etc/gshadow

Client

```
+::::::
```

23.16 /etc/firewall.conf

Server

```
###############################################################################
# You should put this config-file in /etc/arno-iptables-firewall/             #
###############################################################################

# --------------------------- Configuration file ------------------------------
#                       -= Arno's iptables firewall =-
#         Single- & multi-homed firewall script with DSL/ADSL support
#
# (C) Copyright 2001-2007 by Arno van Amersfoort
# Homepage  : http://rocky.eld.leidenuniv.nl/
# Freshmeat : http://freshmeat.net/projects/iptables-firewall/?topic_id=151
# Email     : arnova AT rocky DOT eld DOT leidenuniv DOT nl
#             (note: you must remove all spaces and substitute the @ and the .
#              at the proper locations!)
# -----------------------------------------------------------------------------
# This program is free software; you can redistribute it and/or
# modify it under the terms of the GNU General Public License
# version 2 as published by the Free Software Foundation.

# This program is distributed in the hope that it will be useful, but WITHOUT
# ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
# FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for
# more details.

# You should have received a copy of the GNU General Public License along with
# this program; if not, write to the Free Software Foundation Inc., 59 Temple
```

```
# Place - Suite 330, Boston, MA 02111-1307, USA.
# ------------------------------------------------------------------------


# Location of the iptables-binary (use 'locate iptables' or 'whereis iptables'
# to manually locate it).
# ------------------------------------------------------------------------
IPTABLES="/usr/sbin/iptables"

########################################################################
# External (internet) interface settings                               #
########################################################################

# The external interface(s) that will be protected (and used as internet
# connection). This is probably ppp+ or dsl+ for non-transparent(!) (A)DSL
# modems otherwise it's probably "ethX" (eg. eth0). Multiple interfaces should
# be space separated.
# ------------------------------------------------------------------------
#EXT_IF="ppp+"
EXT_IF="eth0"

# Enable if THIS machines (dynamically) obtains its IP through DHCP (from your
# ISP).
# ------------------------------------------------------------------------
EXT_IF_DHCP_IP=1

# (EXPERT SETTING!) Here you can specify your external(!) subnet(s). You should
# only use this if you for example have a corporate network and/or running a
# DHCP server on your external(!) interface. Home users should normally NOT
# touch this setting. Multiple subnets should be space separated.
# Don't forget to specify a proper subnet masker (eg. /24, /16 or /8)!
# ------------------------------------------------------------------------
EXTERNAL_NET=""

# (EXPERT SETTING!) Here you can specify the IP address used for broadcasts
# on your external subnet. You only need to set this option if you want to use
# the BROADCAST_XXX_NOLOG variables AND you use a non-standard broadcast
# address (not *.255.255.255, *.*.255.255 or *.*.*.255)! So normally leaving
# this empty should work fine. Multiple addresses (if you have more than one
# external interface) should be space separated.
# ------------------------------------------------------------------------
EXT_NET_BCAST_ADDRESS=""

# Enable this if THIS MACHINE is running a DHCP(BOOTP) server for a subnet on
# the external(!) interface. Note that you don't need this for internal
# subnets, as for these nets everything is accepted by default. Don't forget to
# configure the EXTERNAL_NET variable, to make this work.
# ------------------------------------------------------------------------
EXTERNAL_DHCP_SERVER=0


########################################################################
# Internal (LAN) interface settings                                    #
########################################################################

# Specify here your internal network (LAN) interface(s). Multiple(!) interfaces
# should be space separated. Remark this if you don't have any internal network
# interfaces. Note that by default ALL traffic is accepted from these
# interfaces.
```

```
# ------------------------------------------------------------------------
INT_IF="eth1"

# Specify here the internal subnet which is connected to the internal interface
# (INT_IF). For multiple interfaces(!) you can either specify multiple subnets
# here or specify one big subnet for all internal interfaces. Note that this
# variable is mainly used for antispoofing.
# ------------------------------------------------------------------------
INTERNAL_NET="192.168.123.0/24"

# (EXPERT SETTING!) Here you can specify the IP address used for broadcasts
# on your internal subnet. You only need to set this option if you want to use
# the MAC filter AND you use a non-standard broadcast address
# (not *.255.255.255, *.*.255.255 or *.*.*.255)! So normally leaving
# this empty should work fine. Multiple addresses (if you have multiple
# internal nets) should be space separated.
# ------------------------------------------------------------------------
INT_NET_BCAST_ADDRESS="192.168.123.255"

# Uncomment & specify here the location of the file that contains the MAC
# addresses of INTERNAL hosts that are allowed. The MAC addresses should be
# written like 00:11:22:33:44:55
# Note that the last line of this
# file should always contain a carriage-return (enter)!
# ------------------------------------------------------------------------
#MAC_ADDRESS_FILE="/etc/arno-iptables-firewall/mac-addresses"


##########################################################################
# (ADSL) Modem settings                                                  #
#                                                                        #
# The MODEM_xxx options should (only) be used when you have an ((A)DSL)  #
# modem which works with a ppp-connection between the modem and the      #
# host the modem is connected to.                                        #
#                                                                        #
# You can check whether this applies for your (hardware) setup with      #
# 'ifconfig' (a 'ppp' device is shown).                                  #
# This means that if your modem is bridging or an NAT router) or the     #
# network interface the modem is connected to doesn't have an IP, you    #
# should leave the MODEM_xxx options disabled (=default)!                #
##########################################################################

# The physical(!) network interface your ADSL modem is connected to (this is
# not ppp0!).
# ------------------------------------------------------------------------
#MODEM_IF="eth1"

# (optional) The IP of the network interface (MODEM_IF) your ADSL modem is
# connected to (IP shown for the modem interface (MODEM_IF) in 'ifconfig').
# ------------------------------------------------------------------------
#MODEM_IF_IP="10.0.0.150"

# (optional) The IP of your (A)DSL modem itself.
# ------------------------------------------------------------------------
#MODEM_IP="10.0.0.138"

# (EXPERT SETTING!). Here you can specify the hosts/local net(s) that should
# have access to the (A)DSL modem itself (manage modem settings, if supported
# by your modem!). The default setting ("$INTERNAL_NET") allows access from
```

```
# everybody on your LAN.
# ------------------------------------------------------------------------
#MODEM_INTERNAL_NET="$INTERNAL_NET"


##############################################################################
# DMZ (aka DeMilitarized Zone) settings                                      #
##############################################################################

# Put in the following variable the network interfaces that are DMZ-classified.
# You can also use this interface if you want to shield your Wireless network
# from your LAN.
# ------------------------------------------------------------------------
DMZ_IF=""

# Specify here the subnet which is connected to the DMZ interface (DMZ_IF).
# For multiple interfaces(!) you can either specify multiple subnets here or
# specify one big subnet for all DMZ interfaces.
# ------------------------------------------------------------------------
DMZ_NET=""


##############################################################################
# NAT (Masquerade, SNAT, DNAT) settings                                      #
##############################################################################

# Enable this if you want to perform NAT (masquerading) for your internal
# network (LAN) (eg. share your internet connection with your internal
# net(s) connected to eg. INT_IF).
# ------------------------------------------------------------------------
NAT=1

# (EXPERT SETTING!). In case you would like to use SNAT instead of
# MASQUERADING then uncomment and set the IP or IP's here of your static
# external address(es). Note that when multiple IP's are specified, SNAT
# multiroute is enabled (load balancing over multiple external (internet)
# interfaces, check the README file for more info). Note that the order of IP's
# should match the order of interfaces (they belong to) in $EXT_IF!
# ------------------------------------------------------------------------
#NAT_STATIC_IP="193.2.1.1"

# (EXPERT SETTING!). Use this variable only if you want specific subnets or
# hosts to be able to access the internet. When no value is specified, your
# whole internal net will have access. In both cases it's obviously only
# meaningful when NAT is enabled. Note that you can also use this variable if
# you want to use NAT for your DMZ.
# ------------------------------------------------------------------------
NAT_INTERNAL_NET="$INTERNAL_NET"

# NAT TCP/UDP/IP forwards. Forward ports or protocols from the gateway to
# an internal client through (D)NAT. Note that you can also use these
# variables to forward ports to DMZ hosts.
#
# TCP/UDP form:
#       "{SRCIP1,SRCIP2,...:}PORT1,PORT2-PORT3,...>DESTIP1{:port} \
#        {SRCIP3,...:}PORT3,...>DESTIP2:port}"
#
# IP form:
#       "{SRCIP1,SRCIP2,...:}PROTO1,PROTO2,...>DESTIP1 \
```

```
#        {SRCIP3:}PROTO3,PROTO4,...>DESTIP2"
#
# TCP/UDP port forward examples:
# Simple (forward port 80 to internal host 192.168.0.10):
#       NAT_xxx_FORWARD="80>192.168.0.10"
# Advanced (forward port 20 & 21 to 192.168.0.10 and
#           forward from 1.2.3.4 port 81 to 192.168.0.11 port 80:
#       NAT_xxx_FORWARD="20,21>192.168.0.10 1.2.3.4:81>192.168.0.11:80"
#
# IP protocol forward example:
#        (forward protocols 47 & 48 to 192.168.0.10)
#        NAT_IP_FORWARD="47,48>192.168.0.10"
#
# NOTE 1: {:port} is optional. Use it to redirect a specific port to a
#         different port on the internal client.
# NOTE 2: {SRCIPx} is optional. Use it to restrict access for specific source
#         (inet) IP addresses.
# ------------------------------------------------------------------------
NAT_TCP_FORWARD=""
NAT_UDP_FORWARD=""
NAT_IP_FORWARD=""
NAT_TCP_FORWARD="137-139>192.168.123.3 445>192.168.123.3 30000>192.168.123.3 4242>192.168.123.3 4661>192.168.123.3"
NAT_UDP_FORWARD="137-139>192.168.123.3 445>192.168.123.3 30000>192.168.123.3 4242>192.168.123.3 4661>192.168.123.3"

########################################################################
# General settings                                                     #
########################################################################

# Most people don't want to get any firewall logs being spit to the console.
# This option makes the kernel ring buffer only log messages with level
# "panic".
# ------------------------------------------------------------------------
DMESG_PANIC_ONLY=1

# Enable this if you want TOS mangling (RFC) (recommended).
# ------------------------------------------------------------------------
MANGLE_TOS=1

# Enable this if you want to set the maximum packet size via the
# Maximum Segment Size(through MSS field) (recommended).
# ------------------------------------------------------------------------
SET_MSS=1

# Enable this if you want to increase the TTL value by one in the prerouting
# chain. This hides the firewall when performing eg. traceroutes to internal
# hosts.
# ------------------------------------------------------------------------
TTL_INC=0

# (EXPERT SETTING!) Enable this if you want to set the TTL value for packets in
# the OUTPUT & FORWARD chain. Note that this only works with newer 2.6 kernels
# (2.6.14 or better) or patched 2.4 kernels, which have netfilter TTL target
# support. Don't mess with this unless you really know what you are doing!
# ------------------------------------------------------------------------
#PACKET_TTL="64"

# Enable this to resolve names of DNS IP's etc.
# ------------------------------------------------------------------------
RESOLV_IPS=0
```

```
# Enable this to support the IRC-protocol.
# ------------------------------------------------------------------------
USE_IRC=0

# (EXPERT SETTING!). Loosen the forward chain for the external interface(s).
# Enable it to allow the use of protocols like UPnP. Note that it *could* be
# less secure.
# ------------------------------------------------------------------------
LOOSE_FORWARD=0

# (EXPERT SETTING!). Enable this if you want to drop packets originating from a
# private address.
# ------------------------------------------------------------------------
DROP_PRIVATE_ADDRESSES=0

# (EXPERT SETTING!). Protect this machine from being abused for a DRDOS-attack
# ("Distributed Reflection Denial Of Service"-attack). (STILL EXPERIMENTAL!)
# ------------------------------------------------------------------------
DRDOS_PROTECT=0

# Enable this if you want to allow/enable IPv6 traffic. Note that my firewall
# does NOT filter IPv6 traffic (yet), and thus NO checking is performed on it!
# ------------------------------------------------------------------------
IPV6_OVER_IPV4=0

# This option fixes problems with SMB broadcasts when using nmblookup
# ------------------------------------------------------------------------
NMB_BROADCAST_FIX=0

# Set this to 0 to suppress "assuming module is compiled in kernel" messages
# ------------------------------------------------------------------------
COMPILED_IN_KERNEL_MESSAGES=1

# (EXPERT SETTING!). Iptables default policy is DROP. This means that when there
# are no rule(s) available (yet), the packet will be DROPPED. In practice this
# rule only does something while the firewall is starting. Once it's started
# and all rules are in place, the default policy doesn't do anything anymore.
# People that use ie. NFS and let their clients boot from NFS (diskless client
# systems) probably want to disable this option to fix
# "NFS server not responding" etc. errors on their clients.
# ------------------------------------------------------------------------
DEFAULT_POLICY_DROP=1

# (EXPERT SETTING!). (Other) trusted network interfaces for which ALL IP
# traffic should be ACCEPTED. (multiple(!) interfaces should be space
# separated). Be warned that anything TO and FROM these interfaces is allowed
# (ACCEPTED) so make sure it's NOT routable(accessible) from the outside world
# (internet)!
# ------------------------------------------------------------------------
TRUSTED_IF="eth1"

# (EXPERT SETTING!). Put here the (internal) interfaces that should trust
# (accept forward traffic) each other.
# ------------------------------------------------------------------------
INT_IF_TRUST=""

# Location of the custom iptables rules file (if any).
# ------------------------------------------------------------------------
```

```
CUSTOM_RULES="/etc/arno-iptables-firewall/custom-rules"


###############################################################################
# Logging options - All logging is rate limited to prevent log flooding       #
###############################################################################

# Enable logging for explicitly blocked hosts.
# ------------------------------------------------------------------------------
BLOCKED_HOST_LOG=1

# Enable logging for various stealth scans (reliable).
# ------------------------------------------------------------------------------
SCAN_LOG=1

# Enable logging for possible stealth scans (less reliable).
# ------------------------------------------------------------------------------
POSSIBLE_SCAN_LOG=1

# Enable logging for TCP-packets with bad flags.
# ------------------------------------------------------------------------------
BAD_FLAGS_LOG=1

# Enable logging of invalid TCP packets. Keep disabled (0) by default to reduce
# INVALID packets being logged because of lost (legimate) connections. When
# debugging any problems, you should enable it (temporarily)!
# ------------------------------------------------------------------------------
INVALID_TCP_LOG=0

# Enable logging of invalid UDP packets. Keep disabled (0) by default to reduce
# INVALID packets being logged because of lost (legimate) connections. When
# debugging any problems, you should enable it (temporarily)!
# ------------------------------------------------------------------------------
INVALID_UDP_LOG=0

# Enable logging of invalid ICMP packets. Keep disabled (0) by default to reduce
# INVALID packets being logged because of lost (legimate) connections. When
# debugging any problems, you should enable it (temporarily)!
# ------------------------------------------------------------------------------
INVALID_ICMP_LOG=0

# Enable logging of source IP's with reserved addresses.
# ------------------------------------------------------------------------------
RESERVED_NET_LOG=1

# Enable logging of fragmented packets.
# ------------------------------------------------------------------------------
FRAG_LOG=1

# Enable logging of denied local (OUTPUT) connections.
# ------------------------------------------------------------------------------
OUTPUT_DENY_LOG=1

# Enable logging of denied LAN output (FORWARD) connections.
# ------------------------------------------------------------------------------
LAN_OUTPUT_DENY_LOG=1

# Enable logging of denied LAN INPUT connections.
# ------------------------------------------------------------------------------
```

```
LAN_INPUT_DENY_LOG=1

# Enable logging of denied DMZ output (FORWARD) connections.
# ------------------------------------------------------------------------
DMZ_OUTPUT_DENY_LOG=1

# Enable logging of denied DMZ input (FORWARD) connections.
# ------------------------------------------------------------------------
DMZ_INPUT_DENY_LOG=1

# Enable logging of dropped ICMP-request packets (ping).
# ------------------------------------------------------------------------
ICMP_REQUEST_LOG=1

# Enable logging of dropped "other" ICMP packets.
# ------------------------------------------------------------------------
ICMP_OTHER_LOG=1

# Enable logging of normal connection attempts to privileged TCP ports.
# ------------------------------------------------------------------------
PRIV_TCP_LOG=1

# Enable logging of normal connection attempts to privileged UDP ports.
# ------------------------------------------------------------------------
PRIV_UDP_LOG=1

# Enable logging of normal connection attempts to unprivileged TCP ports.
# ------------------------------------------------------------------------
UNPRIV_TCP_LOG=1

# Enable logging of normal connection attempts to unprivileged UDP ports.
# ------------------------------------------------------------------------
UNPRIV_UDP_LOG=1

# Enable logging of normal connection attempts to "other-IP"-protocols (non
# TCP/UDP/ICMP).
# ------------------------------------------------------------------------
OTHER_IP_LOG=1

# Enable logging for ICMP flooding.
# ------------------------------------------------------------------------
ICMP_FLOOD_LOG=1

# Enable logging for not-allowed MAC addresses (if used).
# ------------------------------------------------------------------------
MAC_ADDRESS_LOG=1

# (EXPERT SETTING!). The location of the dedicated firewall log file. When
# enabled the firewall script will also log start/stop etc. info to this file
# as well. Note that in order to make this work, you should also configure
# syslogd to log firewall messages to this file (see LOGLEVEL below for further
# info).
# ------------------------------------------------------------------------
#FIREWALL_LOG="/var/log/firewall"

# (EXPERT SETTING!). Current log-level ("info": default kernel syslog level)
# "debug": can be used to log to /var/log/firewall.log, but you have to configure
# syslogd accordingly (see included syslogd.conf examples).
# ------------------------------------------------------------------------
```

```
LOGLEVEL="debug"

# Put in the following variables which hosts you want to log certain incoming
# connection attempts for.
# TCP/UDP port format (LOG_HOST_xxx_INPUT):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (LOG_HOST_IP_INPUT):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
# ------------------------------------------------------------------------
LOG_HOST_TCP_INPUT=""
LOG_HOST_UDP_INPUT=""
LOG_HOST_IP_INPUT=""

# Put in the following variables which hosts you want to log certain outgoing
# connection attempts for.
# TCP/UDP port format (LOG_HOST_xxx_OUTPUT):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (LOG_HOST_IP_OUTPUT):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
# ------------------------------------------------------------------------
LOG_HOST_TCP_OUTPUT=""
LOG_HOST_UDP_OUTPUT=""
LOG_HOST_IP_OUTPUT=""

# Put in the following variables which services you want to log incoming
# connection attempts for.
# ------------------------------------------------------------------------
LOG_TCP_INPUT=""
LOG_UDP_INPUT=""
LOG_IP_INPUT=""

# Put in the following variables which services you want to log outgoing
# connection attempts for.
# ------------------------------------------------------------------------
LOG_TCP_OUTPUT=""
LOG_UDP_OUTPUT=""
LOG_IP_OUTPUT=""

# Put in the following variable which hosts you want to log incoming connection
# (attempts) for.
# ------------------------------------------------------------------------
LOG_HOST_INPUT=""

# Put in the following variable which hosts you want to log outgoing connection
# (attempts) to.
# ------------------------------------------------------------------------
LOG_HOST_OUTPUT=""


##############################################################################
# /proc based settings (EXPERT SETTINGS!)                                    #
##############################################################################

# Enable for synflood protection (through /proc/.../tcp_syncookies).
# ------------------------------------------------------------------------
SYN_PROT=1
```

```
# Enable this to reduce the ability of others DOS'ing your machine.
# ------------------------------------------------------------------------
REDUCE_DOS_ABILITY=1

# Enable to ignore all ICMP echo-requests (IPv4) on ALL interfaces.
# ------------------------------------------------------------------------
ECHO_IGNORE=0

# Enable to log packets with impossible addresses to the kernel log.
# ------------------------------------------------------------------------
LOG_MARTIANS=0

# Only disable this if you're NOT using forwarding (required for NAT etc.) for
# increased security.
# ------------------------------------------------------------------------
IP_FORWARDING=1

# Enable if you want to accept ICMP redirect messages. Should be set to "0" in
# case of a router.
# ------------------------------------------------------------------------
ICMP_REDIRECT=0

# Enable/modify this if you want to be a able to handle a larger (or smaller)
# number of simultaneous connections. For high traffic machines I recommend to
# use a value of at least 16384 (note that a higher value (obviously) also uses
# more memory).
# ------------------------------------------------------------------------
CONNTRACK=16384

# You may need to enable this to get some internet games to work, but note that
# it's *less* secure.
# ------------------------------------------------------------------------
LOOSE_UDP_PATCH=0

# Enable ECN (Explicit Congestion Notification) TCP flag. Disabled by default,
# as some routers are still not compatible with this.
# ------------------------------------------------------------------------
ECN=0

# Enable to drop connections from non-routable IP's, eg. prevent source
# routing. By default the firewall itself also provides rules against source
# routing. Note than when you use eg. VPN (Freeswan), you should probably
# disable this setting.
# ------------------------------------------------------------------------
RP_FILTER=1

# Protect against source routed packets. Attackers can use source routing to
# generate traffic pretending to be from inside your network, but which is
# routed back along the path from which it came, namely outside, so attackers
# can compromise your network. Source routing is rarely used for legitimate
# purposes, so normally you should always leave this enabled(1)!
# ------------------------------------------------------------------------
SOURCE_ROUTE_PROTECTION=1

# Here we set the local port range (ports from which connections are
# initiated from our site). Don't mess with this unless you really know what
# you are doing!
# ------------------------------------------------------------------------
LOCAL_PORT_RANGE="32768 61000"
```

```
# Here you can change the default TTL used for sending packets. The value
# should be between 10 and 255. Don't mess with this unless you really know
# what you are doing!
# ------------------------------------------------------------------------
DEFAULT_TTL=64

# In most cases pmtu discovery is ok, but in some rare cases (when having
# problems) you might want to disable it.
# ------------------------------------------------------------------------
NO_PMTU_DISCOVERY=0


##############################################################################
# (Transparent) proxy settings (EXPERT SETTINGS!)                            #
##############################################################################
#HTTP_PROXY_PORT="3128"
HTTPS_PROXY_PORT=""
FTP_PROXY_PORT=""
SMTP_PROXY_PORT=""
POP3_PROXY_PORT=""


##############################################################################
# Firewall policies for the LAN (EXPERT SETTINGS!)                           #
##############################################################################

##############################################################################
# LAN_xxx = LAN->localhost(this machine) input access rules                  #
#                                                                            #
# Note that when both LAN_OPEN_xxx & LAN_HOST_OPEN_xxx are NOT used, the     #
# default policy for this chain is accept (unless denied through             #
# LAN_DENY_xxx and/or LAN_HOST_DENY_xxx)!                                    #
##############################################################################

# Enable this to allow for ICMP-requests(ping) from your LAN
# ------------------------------------------------------------------------
LAN_OPEN_ICMP=1

# Put in the following variables the TCP/UDP ports or IP protocols TO
# (remote end-point) which the LAN hosts are permitted to connect to.
# ------------------------------------------------------------------------
LAN_OPEN_TCP=""
LAN_OPEN_UDP=""
LAN_OPEN_IP=""

# Put in the following variables the TCP/UDP ports or IP protocols TO (remote
# end-point) which LAN hosts are NOT permitted to connect to.
# ------------------------------------------------------------------------
LAN_DENY_TCP=""
LAN_DENY_UDP=""
LAN_DENY_IP=""

# Put in the following variables the TCP/UDP ports or IP
# protocols TO (remote end-point) which certain LAN hosts are
# permitted to connect to.
#
# TCP/UDP port format (LAN_INPUT_HOST_OPEN_xxx):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
```

```
#
# IP protocol format (LAN_INPUT_HOST_OPEN_xxx):
#       "host1,host2>proto1,proto2 host3,host4>proto3,proto4 ..."
# ------------------------------------------------------------------------
LAN_HOST_OPEN_TCP=""
LAN_HOST_OPEN_UDP=""
LAN_HOST_OPEN_IP=""

# Put in the following variables the TCP/UDP ports or IP protocols TO (remote
# end-point) which certain LAN hosts are NOT permitted to connect to.
#
# TCP/UDP port format (LAN_INPUT_HOST_DENY_xxx):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (LAN_INPUT_HOST_DENY_xxx):
#       "host1,host2>proto1,proto2 host3,host4>proto3,proto4 ..."
# ------------------------------------------------------------------------
LAN_HOST_DENY_TCP=""
LAN_HOST_DENY_UDP=""
LAN_HOST_DENY_IP=""


##########################################################################
# LAN_INET_xxx = LAN->internet access rules (forward)                    #
#                                                                        #
# Note that when both LAN_INET_OPEN_xxx & LAN_INET_HOST_OPEN_xxx are NOT #
# used, the default policy for this chain is accept (unless denied       #
# through LAN_INET_DENY_xxx and/or LAN_INET_HOST_DENY_xxx)!              #
##########################################################################

# Enable this to allow for ICMP-requests(ping) for LAN->INET
# ------------------------------------------------------------------------
LAN_INET_OPEN_ICMP=1

# Put in the following variables the TCP/UDP ports or IP
# protocols TO (remote end-point) which the LAN hosts are
# permitted to connect to via the external (internet) interface.
# ------------------------------------------------------------------------
LAN_INET_OPEN_TCP=""
LAN_INET_OPEN_UDP=""
LAN_INET_OPEN_IP=""

# Put in the following variables the TCP/UDP ports or IP protocols TO (remote
# end-point) which the LAN hosts are NOT permitted to connect to
# via the external (internet) interface. Examples of usage are for blocking
# IRC (TCP 6666:6669) for the internal network.
# ------------------------------------------------------------------------
LAN_INET_DENY_TCP=""
LAN_INET_DENY_UDP=""
LAN_INET_DENY_IP=""

# Put in the following variables which LAN hosts you want to allow to certain
# hosts/services on the internet. By default all services are allowed.
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
```

```
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple:
#       (Allow port 80 on INET host 1.2.3.4 for all LAN hosts(0/0)):
#       LAN_INET_HOST_OPEN_xxx="0/0>1.2.3.4:80"
# Advanced:
#       (Allow port 20 & 21 on INET host 1.2.3.4 for all LAN hosts(0/0) and
#        allow port 80 on INET host 1.2.3.4 for LAN host 192.168.0.10 (only)):
#       LAN_INET_HOST_OPEN_xxx="0/0>1.2.3.4:20,21 192.168.0.10>80"
#
# IP protocol example:
#       (Allow protocols 47 & 48 on INET host 1.2.3.4 for all LAN hosts(0/0))
#       LAN_INET_HOST_OPEN_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
LAN_INET_HOST_OPEN_TCP=""
LAN_INET_HOST_OPEN_UDP=""
LAN_INET_HOST_OPEN_IP=""

# Put in the following variables which DMZ hosts you want to deny to certain
# hosts/services on the internet.
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Deny port 80 on INET host 1.2.3.4 for all LAN hosts(0/0)):
#       LAN_INET_HOST_DENY_xxx="0/0>1.2.3.4:80"
# Advanced (Deny port 20 & 21 on INET host 1.2.3.4 for all LAN hosts(0/0) and
#           deny port 80 on INET host 1.2.3.4 for LAN host 192.168.0.10 (only)):
#       LAN_INET_HOST_DENY_xxx="0/0>1.2.3.4:20,21 192.168.0.10>1.2.3.4:80"
#
# IP protocol example:
#       (Deny protocols 47 & 48 on INET host 1.2.3.4 for all LAN hosts(0/0)):
#       LAN_INET_HOST_DENY_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
LAN_INET_HOST_DENY_TCP=""
LAN_INET_HOST_DENY_UDP=""
LAN_INET_HOST_DENY_IP=""


##############################################################################
# Firewall policies for the DMZ (EXPERT SETTINGS!)                           #
##############################################################################
```

```
##############################################################################
# DMZ_xxx      = DMZ->localhost(this machine) input access rules             #
##############################################################################

# Enable this to allow ICMP-requests(ping) from the DMZ
# ----------------------------------------------------------------------------
DMZ_OPEN_ICMP=1

# Put in the following variables which DMZ hosts are permitted to connect to
# certain the TCP/UDP ports, IP protocols or ICMP. By default all (local)
# services are blocked for DMZ hosts.
# ----------------------------------------------------------------------------
DMZ_OPEN_TCP=""
DMZ_OPEN_UDP=""
DMZ_OPEN_IP=""

# Put in the following variables which DMZ hosts you want to allow for certain
# services. By default all (local) services are blocked for DMZ hosts.
# TCP/UDP port format (DMZ_HOST_OPEN_TCP & DMZ_HOST_OPEN_UDP):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (DMZ_HOST_OPEN_IP):
#       "host1,host2>proto1,proto2 host3,host4>proto3,proto4 ..."
# ----------------------------------------------------------------------------
DMZ_HOST_OPEN_TCP=""
DMZ_HOST_OPEN_UDP=""
DMZ_HOST_OPEN_IP=""


##############################################################################
# INET_DMZ_xxx = Internet->DMZ access rules (forward)                        #
#                                                                            #
# Note that when both INET_DMZ_OPEN_xxx & INET_DMZ_HOST_OPEN_xxx are NOT     #
# used, the default policy for this chain is accept (unless denied           #
# through INET_DMZ_DENY_xxx and/or INET_DMZ_HOST_DENY_xxx)!                  #
##############################################################################

# Enable this to make the default policy allow for ICMP(ping) for INET->DMZ
# ----------------------------------------------------------------------------
INET_DMZ_OPEN_ICMP=0

# Put in the following variables which INET hosts are permitted to connect to
# certain the TCP/UDP ports or IP protocols in the DMZ.
# ----------------------------------------------------------------------------
INET_DMZ_OPEN_TCP=""
INET_DMZ_OPEN_UDP=""
INET_DMZ_OPEN_IP=""

# Put in the following variables which INET hosts are NOT permitted to connect
# to certain the TCP/UDP ports or IP protocols in the DMZ.
# ----------------------------------------------------------------------------
INET_DMZ_DENY_TCP=""
INET_DMZ_DENY_UDP=""
INET_DMZ_DENY_IP=""

# Put in the following variables which INET hosts you want to allow to certain
# hosts/services on the DMZ net. By default all services are allowed.
#
# TCP/UDP form:
```

```
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Allow port 80 on DMZ host 1.2.3.4 for all INET hosts(0/0)):
#       INET_DMZ_HOST_OPEN_xxx="0/0>1.2.3.4:80"
# Advanced (Allow port 20 & 21 on DMZ host 1.2.3.4 for all INET hosts(0/0) and
#           allow port 80 on DMZ host 1.2.3.4 for INET host 5.6.7.8 (only)):
#       INET_DMZ_HOST_OPEN_xxx="0/0>1.2.3.4:20,21 5.6.7.8>1.2.3.4:80"
#
# IP protocol example:
#       (Allow protocols 47 & 48 on INET host 1.2.3.4 for all DMZ hosts )
#       INET_DMZ_HOST_OPEN_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
INET_DMZ_HOST_OPEN_TCP=""
INET_DMZ_HOST_OPEN_UDP=""
INET_DMZ_HOST_OPEN_IP=""

# Put in the following variables which INET hosts you want to deny to certain
# hosts/services on the DMZ net.
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Deny port 80 on DMZ host 1.2.3.4 for all INET hosts(0/0)):
#       INET_DMZ_HOST_DENY_xxx="0/0>1.2.3.4:80"
# Advanced (Deny port 20 & 21 on DMZ host 1.2.3.4 for all INET hosts(0/0) and
#           deny port 80 on DMZ host 1.2.3.4 for INET host 5.6.7.8 (only)):
#       INET_DMZ_HOST_DENY_xxx="0/0>1.2.3.4:20,21 5.6.7.8>1.2.3.4:80"
#
# IP protocol example:
#       (Deny protocols 47 & 48 on DMZ host 1.2.3.4 for all INET hosts):
#       INET_DMZ_HOST_DENY_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
INET_DMZ_HOST_DENY_TCP=""
INET_DMZ_HOST_DENY_UDP=""
INET_DMZ_HOST_DENY_IP=""


##############################################################################
# DMZ_INET_xxx = DMZ->internet access rules (forward)                        #
```

```
#                                                                              #
# Note that when both DMZ_INET_OPEN_xxx & DMZ_INET_HOST_OPEN_xxx are NOT       #
# used, the default policy for this chain is accept (unless denied             #
# through DMZ_INET_DENY_xxx and/or DMZ_INET_HOST_DENY_xxx)!                    #
################################################################################

# Enable this to make the default policy allow for ICMP(ping) for DMZ->INET
# ------------------------------------------------------------------------------
DMZ_INET_OPEN_ICMP=1

# Put in the following variables the TCP/UDP ports or IP
# protocols TO (remote end-point) which the DMZ hosts are
# permitted to connect to via the external (internet) interface.
# ------------------------------------------------------------------------------
DMZ_INET_OPEN_TCP=""
DMZ_INET_OPEN_UDP=""
DMZ_INET_OPEN_IP=""

# Put in the following variables the TCP/UDP ports or IP protocols TO (remote
# end-point) which the DMZ hosts are NOT permitted to connect to
# via the external (internet) interface. Examples of usage are for blocking
# IRC (TCP 6666:6669) for the internal network.
# ------------------------------------------------------------------------------
DMZ_INET_DENY_TCP=""
DMZ_INET_DENY_UDP=""
DMZ_INET_DENY_IP=""

# Put in the following variables which DMZ hosts you want to allow to certain
# hosts/services on the internet. By default all services are allowed.
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Allow port 80 on INET host 1.2.3.4 for all DMZ hosts(0/0)):
#       DMZ_INET_HOST_OPEN_xxx="0/0>1.2.3.4:80"
# Advanced (Allow port 20 & 21 on INET host 1.2.3.4 for all DMZ hosts(0/0) and
#           allow port 80 on INET host 1.2.3.4 for DMZ host 5.6.7.8 (only)):
#       DMZ_INET_HOST_OPEN_xxx="0/0>1.2.3.4:20,21 5.6.7.8>1.2.3.4:80"
#
# IP protocol example:
#       (Allow protocols 47 & 48 on INET host 1.2.3.4 for all DMZ hosts):
#       DMZ_INET_HOST_OPEN_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------------
DMZ_INET_HOST_OPEN_TCP=""
DMZ_INET_HOST_OPEN_UDP=""
DMZ_INET_HOST_OPEN_IP=""

# Put in the following variables which DMZ hosts you want to deny to certain
# hosts/services on the internet.
```

```
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Deny port 80 on INET host 1.2.3.4 for all DMZ hosts(0/0)):
#       DMZ_INET_HOST_DENY_xxx="0/0>1.2.3.4:80"
# Advanced (Deny port 20 & 21 on INET host 1.2.3.4 for all DMZ hosts(0/0) and
#           deny port 80 on INET host 1.2.3.4 for DMZ host 5.6.7.8 (only)):
#       DMZ_INET_HOST_DENY_xxx="0/0>1.2.3.4:20,21 5.6.7.8>1.2.3.4:80"
#
# IP protocol example:
#       (Deny protocols 47 & 48 on INET host 1.2.3.4 for all DMZ hosts(0/0)):
#       DMZ_INET_HOST_DENY_IP="0/0>1.2.3.4:47,48"
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
DMZ_INET_HOST_DENY_TCP=""
DMZ_INET_HOST_DENY_UDP=""
DMZ_INET_HOST_DENY_IP=""


############################################################################
# DMZ_LAN_xxx  = DMZ->LAN access rules (forward)                           #
############################################################################

# Enable this to make the default policy allow for ICMP(ping) for DMZ->LAN
# ------------------------------------------------------------------------
DMZ_LAN_OPEN_ICMP=0

# Put in the following variables which DMZ hosts you want to allow to certain
# hosts/services on the LAN (net).
#
# TCP/UDP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:port \
#        SRCIP3,...>DESTIP2:port"
#
# IP form:
#       "SRCIP1,SRCIP2,...>DESTIP1:protocol \
#        SRCIP3,...>DESTIP2:protocol"
#
# TCP/UDP examples:
# Simple (Allow port 80 on LAN host 1.2.3.4 for all DMZ hosts(0/0)):
#       DMZ_LAN_HOST_OPEN_xxx="0/0>1.2.3.4:80"
# Advanced (Allow port 20 & 21 on LAN host 1.2.3.4 for all DMZ hosts (0/0) and
#           allow port 80 for DMZ host 5.6.7.8 (only) on LAN host
#           1.2.3.4):
#       DMZ_LAN_HOST_OPEN_xxx="0/0>1.2.3.4:20,21 5.6.7.8>1.2.3.4:80"
#
# IP protocol example:
#       (Allow protocols 47 & 48 on LAN host 1.2.3.4 for all DMZ hosts(0/0)):
#       DMZ_LAN_HOST_OPEN_IP="0/0>1.2.3.4:47,48"
```

```
#
# NOTE 1: If no SRCIPx is specified, any source host is used
# NOTE 2: If no DESTIPx is specified, any destination host is used
# NOTE 3: If no port is specified, any port is used
# ------------------------------------------------------------------------
DMZ_LAN_HOST_OPEN_TCP=""
DMZ_LAN_HOST_OPEN_UDP=""
DMZ_LAN_HOST_OPEN_IP=""


##########################################################################
# Firewall policies for the external (inet) interface (default policy = drop) #
##########################################################################

# Put in the following variable which hosts (subnets) you want have full access
# via your internet (EXT_IF) connection(!). This is especially meant for
# networks/servers which use NIS/NFS, as these protocols require all ports
# to be open.
# NOTE: Don't mistake this variable with the one used for internal nets.
# ------------------------------------------------------------------------
FULL_ACCESS_HOSTS=""

# Enable this to make the default policy allow for ICMP(ping) for INET access
# ------------------------------------------------------------------------
OPEN_ICMP=0

# Put in the following variables which ports or IP protocols you want to leave
# open to the whole world.
# ------------------------------------------------------------------------
OPEN_TCP="22 123 80 443 3000 8080 3128 32000"
OPEN_UDP="22 123 80 443 3000 8080 3128 32000"
OPEN_IP=""

# Put in the following variables the TCP/UDP ports you want to DENY(DROP) for
# everyone (and logged). Also use these variables if you want to log connection
# attempts to these ports from everyone (also trusted/full access hosts).
# In principle you don't need these variables, as everything is already blocked
# (denied) by default, but just exists for consistency.
# ------------------------------------------------------------------------
DENY_TCP=""
DENY_UDP=""

# Put in the following variables which ports you want to DENY(DROP) for
# everyone but NOT logged. This is very useful if you have constant probes on
# the same port(s) over and over again (code red worm) and don't want your logs
# flooded with it.
# ------------------------------------------------------------------------
DENY_TCP_NOLOG=""
DENY_UDP_NOLOG=""

# Put in the following variables the TCP/UDP ports you want to REJECT (instead
# of DROP) for everyone (and logged).
# ------------------------------------------------------------------------
REJECT_TCP=""
REJECT_UDP=""

# Put in the following variables the TCP/UDP ports you want to REJECT (instead
# of DROP) for everyone but NOT logged.
# ------------------------------------------------------------------------
```

```
REJECT_TCP_NOLOG=""
REJECT_UDP_NOLOG=""

# Put in the following variables which hosts you want to allow for certain
# services.
# TCP/UDP port format (HOST_OPEN_TCP & HOST_OPEN_UDP):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (HOST_OPEN_IP):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
#
# ICMP protocol format (HOST_OPEN_ICMP):
#       "host1 host2 ...."
# ------------------------------------------------------------------------
HOST_OPEN_TCP=""
HOST_OPEN_UDP=""
HOST_OPEN_IP=""
HOST_OPEN_ICMP=""

# Put in the following variables which hosts you want to DENY(DROP) for certain
# services (and logged).
# to DENY(DROP) for certain hosts.
# TCP/UDP port format (HOST_DENY_TCP & HOST_DENY_UDP):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (HOST_DENY_IP):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
#
# ICMP protocol format (HOST_DENY_ICMP):
#       "host1 host2 ...."
# ------------------------------------------------------------------------
HOST_DENY_TCP=""
HOST_DENY_UDP=""
HOST_DENY_IP=""
HOST_DENY_ICMP=""

# Put in the following variables which hosts you want to DENY(DROP) for certain
# services but NOT logged.
# TCP/UDP port format (HOST_DENY_xxx_NOLOG):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (HOST_DENY_IP_NOLOG):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
#
# ICMP protocol format (HOST_DENY_ICMP_NOLOG):
#       "host1 host2 ...."
# ------------------------------------------------------------------------
HOST_DENY_TCP_NOLOG=""
HOST_DENY_UDP_NOLOG=""
HOST_DENY_IP_NOLOG=""
HOST_DENY_ICMP_NOLOG=""

# Put in the following variables which hosts you want to REJECT (instead of
# DROP) for certain TCP/UDP ports.
# TCP/UDP port format (HOST_REJECT_xxx):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
# ------------------------------------------------------------------------
HOST_REJECT_TCP=""
HOST_REJECT_UDP=""
```

```
# Put in the following variables which hosts you want to REJECT (instead of
# DROP) for certain services but NOT logged.
# TCP/UDP port format (HOST_REJECT_xxx_NOLOG):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
# ------------------------------------------------------------------------------
HOST_REJECT_TCP_NOLOG=""
HOST_REJECT_UDP_NOLOG=""

# Put in the following variables which services THIS machine is NOT
# permitted to connect TO (remote end-point) via the external (internet)
# interface. For example for blocking IRC (tcp 6666:6669).
# ------------------------------------------------------------------------------
DENY_TCP_OUTPUT=""
DENY_UDP_OUTPUT=""
DENY_IP_OUTPUT=""

# Put in the following variables to which hosts THIS machine is NOT
# permitted to connect TO for certain services (remote end-point)
# via the external (internet) interface. In principle you can also
# use this to put your machine in a "virtual-DMZ" by blocking all traffic
# to your local subnet.
# TCP/UDP port format (HOST_DENY_TCP_OUTPUT & HOST_DENY_UDP_OUTPUT):
#       "host1,host2>port1,port2 host3,host4>port3,port4 ..."
#
# IP protocol format (HOST_DENY_IP_OUTPUT):
#       "host1,host2>proto1,proto2 host3,host4>proto4,proto4 ..."
# ------------------------------------------------------------------------------
HOST_DENY_TCP_OUTPUT=""
HOST_DENY_UDP_OUTPUT=""
HOST_DENY_IP_OUTPUT=""

# Put in the following variable which TCP/UDP ports you don't want to
# see broadcasts from (ie. DHCP (67/68) on your EXTERNAL interface. Note that
# to make this properly work you also need to set "EXTERNAL_NET"!
# ------------------------------------------------------------------------------
BROADCAST_TCP_NOLOG=""
#BROADCAST_UDP_NOLOG="67 68"

# Put in the following variable which hosts you want to block (blackhole,
# dropping every packet from the host).
# ------------------------------------------------------------------------------
BLOCK_HOSTS=""

# Uncomment & specify here the location of the file that contains a list of
# hosts(IP's) that should be BLOCKED. IP ranges can (only) be specified as
# w.x.y.z1-z2 (ie. 192.168.1.10-15). Note that the last line of this file
# should always contain a carriage-return (enter)!
# ------------------------------------------------------------------------------
#BLOCK_HOSTS_FILE="/etc/arno-iptables-firewall/blocked-hosts"
```

23.17 /usr/local/etc/backup.conf

Client

```
##############################
```

```
# C O N F I G U R A T I O N #
#############################
# Backup v3.0 by loop <loop@foc.dyndns.org> || http://foc.neoartis.org
#
# Check each option patiently. All of them are explained.
# 0 = off; 1 = on

# CAUTION CAUTION
# The references to directories must always be the real PATH to the directory,
# NEVER put a link that points to the real directory in the variables!!

# LANGUAGE
# Set the language for the script to use
# Options: en (english), es (spanish)
lang=en

# COMPRESSION
# Set the binary to use for compression, 'gzip' or 'bzip2'
# Options: gzip, bzip2
#compress=gzip
compress=bzip2

# ENCRYPTION
# Do you want to encrypt (blowfish) the backup files? [0|1]
# To decrypt them later, run:
# $openssl enc -bf -d -k "$encryptpass" -in file.bf -out file
encrypt=0
# PATH to 'openssl' (e.g.: /usr/bin/openssl)
openssl=/usr/bin/openssl
# Key to encrypt with
encryptpass=""

# LOGS
# Directory where we save the stdout log (ej. /var/log/backup, /usr/local/backup/log)
# Tip: use 'logrotate' or any other script to rotate/delete them, I will not add an
#      option to disable them. :P
backupdirlog=/usr/local/backup/log

# NOTIFICATION
# Do you want to receive a notification when I finish the backup process? [0|1]
# NB: you are only defining if you will be notified on a successful backup; error logs
# are always sent, so the PATH to 'mail' and the notified person are required.
notificacion=1
# PATH to 'mail'
mailbin=/bin/mail
# Who will be notified? (e.g.: root, backupadmin@domain.com)
notificado=root


#
# BACKUPS DESTINATION
#

# SAMBA
# Shall I save the backups using SAMBA? [0|1]
samba=0
# PATH to 'smbmount' (e.g.: /usr/bin/smbmount)
smbmount=/usr/bin/smbmount
# Remote machine and resource (e.g.: \\\\server\\resource)
```

```
recurso=
# User to mount the resource as
smbuser=
# User's password
smbpass=
# Where do you want to mount the resource ? (e.g.: /mnt/server)
# NB: This directory MUST be created previously, it won't be created!
montaje=
# OPTIONAL: Subdir within the remote resource where backups should be stored
# NB: This directory MUST be created previously, it won't be created!
subdir=

# TAPE
# Do you want to save your backups on tape (dat)? [0|1]
dat=0
# PATH to 'mt-st' (e.g.: /bin/mt-st)
mtst=/bin/mt-st
# Tape device (e.g.: /dev/st0)
datdev=
# Do you want to remove this backup from HD ($destdir/$fecha) after saving it? [0|1]
borradatbak=0

# CDROM
# Do you want to record your backups on CDROM? [0|1]
cd=0
# Specify CDROM's size in KBytes
# 74" <=> 650 MB <=> 665600 KBytes
# 80" <=> 700 MB <=> 716800 KBytes
cdcap=665600
# PATH to 'cdrecord' and 'mkisofs'
# (e.g.: /usr/bin/cdrecord, /usr/bin/mkisofs)
cdrecord=/usr/bin/cdrecord
mkisofs=/usr/bin/mkisofs
# CD Burner device: scsibus,target,lun (e.g.: 0,3,0)
cddev=
# Recording Speed (e.g.: 8, 12, 32...)
velocidad=
# Directory where the ISO image should be created (e.g.: /tmp, /mnt/grabar, /mnt/iso)
# NB: This directory MUST be created previously, it won't be created!
isodir=
# Do you want to remove the ISO image from the HD after recording it? [0|1]
borraiso=0
# Do you want to remove this backup from HD ($destdir/$fecha) after recording it? [0|1]
borracdbak=0
# In case it's a rewritable CD, should we blank it before recording? [0|1]
cdrw=0
# If cdrw=1, specify how do you want to blank your CDRW: "fast", if you want
# to do it in a faster way, or "disk" for a complete blanking process.
# It's recommended to do a complete blanking from time to time.
borradocd=fast

# ISO ONLY
# Do you only want to make an ISO image of the current backup? [0|1]
# Maybe you only want to make an ISO image of the current backup, deleting from harddisk
# the copy and keeping only the ISO image in a file.
# NB: You have to specify the PATH to 'mkisofs' ($mkisofs) and the directory where the
#     ISO would be created ($isodir) in the previous section, CDROM.
soloiso=0
# OPTIONAL: Specify a command (with absolute PATH to it) for making something with the ISO.
```

```
#           Use the expression "isomsg" (NOT $isoimg!) to reference the ISO file created.
#           e.g.: "/usr/bin/gzip -9 isoimg" or "/usr/bin/bzip2 -9 isoimg" => compress ISO
#                 "/usr/local/bin/cdrecord dev=0,3,0 speed=4 -eject isoimg" => burn ISO
soloisocmd=""

# FTP
# Do you want to transfer the backup by FTP to another machine? [0|1]
# NB: This option doesn't apply if $soloiso=1
ftp=0
# Remote machine
ftpserver=
# User to login as
ftpuser=
# User's password
ftppass=
# Do you want to remove this backup from HD ($destdir/$fecha) after transfer it? [0|1]
borraftpbak=0

# SCP
# Do you want to tranfer the backup by SCP to another machine? [0|1]
# NB: This option doesn't apply if $soloiso=1
# WARNING! WARNING!
# You must have a proper configuration of ssh with identity-based authentication.
# References:  http://www.hackinglinuxexposed.com/articles/20021211.html (3 parts)
#              http://www.linuxfocus.org/English/January2003/article278.shtml
#              http://www.stearns.org/ssh-keyinstall/
#              http://www.gentoo.org/proj/en/keychain.xml
scp=0
# Remote machine
scpserver=
# User to login as
scpuser=
# Directorio de la maquina remota donde se copiara el backup (ej. /home/backupuser)
scpdir=
# Do you want to remove this backup from HD ($destdir/$fecha) after transfer it? [0|1]
borrascpbak=0

# LOCAL HARD DISK
# In case you're using SAMBA (samba=1), forget about this variable, because
# it won't be used, the destination directory is the one where you selected
# to mount the remote resource.
# In any other case, it's NECESSARY TO SPECIFY IT!
# Destination directory of backup files (e.g.: /mnt/backup)
# NB: This directory MUST be created previously, it won't be created!
destdir=/mnt/local/data01


#
# NUMBER OF OLD BACKUPS TO KEEP
#

# Assign 0 to the variable to keep none.
# Assign a negative number or characters to the variable to keep all of them.
# Assign a positive number to the variable and this number of backups will be kept. ;P

# Number of old backups to keep on harddisk or SAMBA resource
guardabackups=3
# Number of old backups to keep on the remote FTP server
guardabackupsftp=2
```

```
# Number of old backups to keep on the remote SCP (SSH) server
guardabackupsscp=2

#
# BACKUP CONTENT
#

# /boot copy [0|1]
boot=1

# /etc copy [0|1]
etc=1

# /usr/local/etc copy [0|1]
localetc=1

# /usr/local copy [0|1]
localall=1

# Crontab copy [0|1]
crontab=1
# Crontab directory (e.g.: /var/spool/cron/crontabs)
crontabdir=/var/spool/cron/crontabs

# System log copy [0|1]
syslog=1
# System log's directory (e.g.: /var/log)
syslogdir=/var/log

# Kernel configuration (/usr/src/linux/.config) [0|1]
kernel=1

# Installed packages listing (tgz, rpm, deb) [0|1]
paquetes=1

# System information (devices, partitions, processes, routes...) [0|1]
sysinfo=1

# Root's home backup [0|1]
root=1
# Specify root's home (e.g.: /root)
rootdir=/root
# Exclude some subdirectories from $rootdir? [0|1]
excluirroot=0
# Specify the absolute path to the directories that are to be excluded,
# separated by spaces.
# NB: put them between inverted commas!
excluirrootdirs=""


# Copy all the homes under your homes' directory [0|1]
homeall=1
# Specify the directory where users' homes are stored (e.g.: /home)
homedir=/home
# Exclude some subdirectories from $homedir ? [0|1]
excluirhomeall=1
# Specify the absolute path to the directories that are to be excluded,
# separated by spaces.
# NB: put them between inverted commas!
```

```
excluirhomealldirs="/home/ftp /home/httpd"

# Selective copy of homes [0|1]
homesel=0
# Specify the filename of the backup's tarball (.tar.gz) without its extension,
# ($userN) and the absolute path to the home directory of each user ($userdirN).
# Continue the series to add more users:
# userN=
# userdirN=
# user(N+1)=
# userdir(N+1)=
# user(N+2)=
# userdir(N+2)=
# etc...
user1=
userdir1=
user2=
userdir2=
# And now specify how many users you have defined
usernum=2

# CHROOT jail copy [0|1]
chroot=0
# CHROOT's directory (e.g.: /chroot)
chrootdir=/chroot

# Do you want to export your PGP keys to a file? [0|1]
pgp=0
# PATH to 'gpg' (e.g.: /usr/local/bin/gpg)
gpg=/usr/local/bin/gpg

# Mail copy [0|1]
mail=1
# Mail's spool directory (e.g.: /var/spool/mail)
maildir=/var/spool/mail

# MailScanner copy [0|1]
mailscanner=0
# MailScanner's main directory (e.g.: /opt/Mailscanner-4.11)
mailscannerdir=/opt/MailScanner
# Do you want to backup the MailScanner's quarantine directory? [0|1]
mailscanner_quarantine=0
# MailScanner's quarantine directory (e.g.: /var/spool/MailScanner/quarantine)
mailscanner_quarantinedir=/var/spool/MailScanner/quarantine
# Do you want to backup the MailScanner's archive directory? [0|1]
# NB: read the MailScanner's configuration file comments about archiving mail!
mailscanner_archive=0
# MailScanner's archive directory (e.g.: /var/spool/MailScanner/archive)
mailscanner_archivedir=/var/spool/MailScanner/archive

# Do you want to copy MySQL's configuration? (/etc/my.cnf) [0|1]
mysqlcfg=0
# Do you want to copy ALL MySQL databases? [0|1]
mysqldb=0
# PATH to 'mysqldump' (e.g.: /usr/local/mysql/bin/mysqldump)
mysqldump=/usr/local/mysql/bin/mysqldump
# MySQL user (e.g.: root or any other with unlimited access)
mysqluser=root
# Specified user's password
```

```
mysqlpass=

# Apache copy [0|1]
apache=1
# Apache's main directory (e.g.: /usr/local/apache)
apachedir=/usr/local/apache
# Exclude some subdirectories from de $apachedir? [0|1]
excluirapache=0
# Specify the absolute path to the directories that are to be excluded,
# separated by spaces.
# NB: put them between inverted commas!
excluirapachedirs=""
# Apache's DocumentRoot directory (e.g.: /var/www, /home/httpd)
# NB: Specify it ONLY if it isn't under Apache's main directory.
apachedocs=
# Exclude some subdirectories from $apachedocs? [0|1]
excluirapachedocs=0
# Specify the absolute path to the directories that are to be excluded,
# separated by spaces.
# NB: put them between inverted commas!
excluirapachedocsdirs=""

# php.ini copy [0|1]
php=1
# Where is php.ini ? (e.g.: /usr/local/lib)
phpdir=/usr/local/lib

# SQUID copy [0|1]
squid=0
# SQUID's directory (e.g.: /usr/local/squid)
squiddir=/usr/local/squid
# Exclude some subdirectories from $squiddir? [0|1]
excluirsquid=1
# Specify the absolute path to the directories that are to be excluded,
# separated by spaces.
# NB: put them between inverted commas!
excluirsquiddirs="/usr/local/squid/cache"

# (UCD/NET) SNMP copy [0|1]
snmp=0
# SNMP's configuration files directory (e.g.: /usr/local/share/snmp)
snmpdir=/usr/local/share/snmp

# MRTG copy [0|1]
mrtg=0
# MRTG's main directory (e.g.: /usr/local/mrtg-2)
mrtgdir=/usr/local/mrtg-2
# MRTG's log directory (e.g.: /var/log/mrtg)
# NB: Specify it ONLY if it isn't under MRTG's main directory.
mrtglogs=

# Majordomo copy [0|1]
majordomo=0
# Majordomo's directory (e.g.: /usr/local/majordomo-1.94.5)
majordomodir=/usr/local/majordomo-1.94.5

# Mailman copy [0|1]
mailman=0
# Mailman's directory (e.g.: /home/mailman, /usr/local/mailman)
```

```
mailmandir=/home/mailman

# AWStats cache copy [0|1]
awstats=0
# AWStats' cache directory (e.g.: /var/cache/awstats)
awstatsdir=/var/cache/awstats

# Nagios copy (before called NetSaint) [0|1]
nagios=0
# Nagios' directory (e.g.: /usr/local/nagios)
nagiosdir=/usr/local/nagios

# CUSTOM DIRECTORIES
# Do you want to backup another directories apart from those specified above? [0|1]
otrosdirs=0
# Specify the filename of the backup's tarball (.tar.gz) without its extension,
# ($otrodN) and the path to each directory ($otrodirN).
# Continue the series to add more directories:
# otrodN=
# otrodirN=
# otrod(N+1)=
# otrodir(N+1)=
# otrod(N+2)=
# otrodir(N+2)=
# etc...
otrod1=
otrodir1=
otrod2=
otrodir2=
# And now specify how much directories you have defined
otrosdnum=2

# Do you want to list another directories' contents? [0|1]
otroslists=0
# Specify the filename of the backup's tarball (.tar.gz) without its extension,
# ($otroN) and the path to the directory ($otrolistN).
# Continue the series to add more directories:
# otrolN=
# otrolistN=
# otrol(N+1)=
# otrolist(N+1)=
# otrol(N+2)=
# otrolist(N+2)=
# etc...
otrol1=
otrolist1=
otrol2=
otrolist2=
# And now specify how much directories you have defined
otroslnum=2

# EOF
```