MyFirewall-(2 Interfaces)

```
#!/bin/bash
# Copyright (c) 2002 Pierre Burri
# MyFirewall is free for personal use only.
# Use this firewall at your own risks, I am NOT responsable if someone
# is able to break through it and corrupt your system.
# You have been warned!
#
# File Name : /etc/init.d/MyFirewall
# Version   : 1.01-elop
# Author    : Pierre Burri, can be reached at pierre@globeall.de
#           This firewall is my own "soup" but inspired from the book
#           "Das Firewall Buch" from Wolfgang Barth, "Linux Firewalls"
#           second edition from Robert L. Ziegler, many articles and
#            my first firewall with ipchains.
# Date      : 19-May-2001
# Release   : 05-Jun-2001  added forwarding for ICMP
#             09-Jun-2001  added possibility to administer the server from
#                     a remote host (ssh)
#           07-Jul-2001  ssh uses now only unprivileged ports
#           10-Jul-2001  removes module ipchains if necessary (SuSE 7.2)
#                     change grep to "inet addr" (SuSE 7.2)
#                     added transparent proxy
#           12-Jul-2001  use REDIRECT for transparent proxying
#                     "iptables -t nat -F" added when the firewall is
#                     stopped
#           02-Nov-2001  added tests for icmp_xxx because some kernel
#                     parameters have disappeared with SuSE 7.3
#           15-Nov-2001  added second ethernet card for ADSL,
#                     added a few new local variables
#           24-Nov-2001  added a test if the script is run in a terminal
#           12-Dez-2001  added MSS (Max Segment Size) correction for ADSL
#                     in the FORWARD chain
#           20-Dez-2001  cleanup unnecessary test lines
#                     added a many more comments, the new chains www_for,
#                     renamed logdropsyn in logdropopen, echo-request in
#                          INPUT but with burst-limit, doesn't log netbios
#                          packets anymore etc...
#             3-Jan-2002  service auth added in OUTPUT filter.
#                          TCP flags check added. New chains tcp_flags,
#                          spoofed_src_ip, spoofed_dst_ip, icpm_in &
#                          icmp_out added. log end of nat table.
#                     Replaced com_out and com_for with www_serv.
#           11-Apr-2002  - changed the grep of the IP Addr because of
#                           english (addr) and german (Adr).
#                          - because of SuSE 8.0 added variables if_config,
#                           net_stat and iptables
#                     - added variable allow_smtp & allow_http for more
#                          flexibility and test purposes
#            15-Apr-2002  - removed eth1 for DSL.
```

```
#                       with pppoe, only ppp0 is used and not eth1.
#                     - added variable allow_smtp_test
#          7-Jun-2002  - fixed ping problem.
#          8-Jun-2002  - added possibility of a permanent ssh entry.
#                     - added variable forwarding for the possibility
#                       to disable forwarding/masquerading functions.
#                     - all lan can be disabled.
#          9-Jun-2002  - fixed bug with ssh remote entry.
#         20-Jun-2002  - added variables allow_pop3 and allow_ftp for
#                       for pop3 server und ftp server.
#                - for tests purposes added variables inside_C_lan
#                       & allow_cups_b.
#         16-Jul-2002  - added Time Server entry and variable
#
# Usage    : /etc/init.d/MyFirewall cmd [ext-IF] [dis|ena|IP-Address]
#
#        1. param: cmd  = start or stop or restart or status.
#        2. param: ext-IF = ppp0 for ADSL/Modem, ippp0 - ipppn for ISDN.
#        3. param: ssh-entry. If there is no 3. parameter or 3. parameter
#                  is "dis", then ssh from outside is disabled (default).
#                  If 3. argument is "ena", then ssh from outside is
#                  enabled. If 3. parameter is an IP-Address, then only
#                  this IP-Address can enter through the firewall.
#
#        MyFirewall should be called by  /etc/ppp/ip-up.local
#        don't forget to make it execubale: chmod 755 /etc/ppp/ip-up.local
#
#         ip-up.local should have the following lines:
#
#        #!/bin/bash
#        /etc/init.d/MyFirewall restart $1
#
#        It is probably a good idea, especially if you run a proxy like
#        squid, to start as well a cache DNS Server (bind9 or bind8)
#        beside the firewall. Don't forget to put the DNS Servers of your
#        ISP (Provider) in the configuration file of  /etc/named.conf
#         (the following example is for T-DSL):
#
#         forwarders { 217.230.170.127; 194.25.2.129; };
#
#        and configure carefully who can access your name server, eg.:
#
#         allow-query { 127.0.0.1; 192.168.10.0/24; };
#
```

```
#-----------------------------------------------------------------
# Script Result variables
# The following two variables were taken from /etc/rc.status
if [ $TERM = "linux" -o $TERM = "xterm" ]
then
  rc_done="\015\033[80C\033[10D\033[1;32mdone\033[m\017"
  rc_failed="\015\033[80C\033[10D\033[1;31mfailed\033[m\017"
else
  rc_done="done"
  rc_failed="failed"
fi
# NOTE: Variables with *** in comments are have to be adapted
# ========================================
# the following 3 variables have to be eventually adapded if "ifconfig" or
# "netstat" or "iptables" are not in the same path on your linux distribution.
# These are set for SuSE Linux 8.0

if_config=/sbin/ifconfig
net_stat=/bin/netstat
iptables=/usr/sbin/iptables

# Definition of local variables.
# --------------------

# *** "ssh_rip" allow to control if a remote connection through SSH to this
# host is possible. The possible values are "dis" (disabled) or "ena"
# (enabled). The value of "ssh_rip" is overwritten, wenn an IP-Address is given
# as a third paramater at the start of the firewall.
ssh_rip=ena

# *** do you want to allow access to your Web server (Apache) from outside?
# default = no
allow_http=no

# *** do you want to allow access to your Mail server (Sendmail, Postfix, Qmail) from
# outside?
# default = no
allow_smtp=no

# the following variable is only for SMTP-Test purposes
allow_smtp_test=no

# *** do you want to allow access to your POP-3 server (qpopper)
# from outside?
# default = no
allow_pop3=no

# *** do you want to allow access to your FTP server from outside?
# default = no
allow_ftp=no
# is your firewall inside of a class C lan? MyFirewall is thought to
# protect a lan from the Internet. But, mainly for test purposes, it is
# possible to setup MyFirewall to protect a single host inside of a
# class C lan.
```

```
# If you set inside_C_lan=yes, probably def_ext_int will be = eth0,
# lan1 & lan2 will be disable and forwarding will be = no.
# Be sure you understand what you are doing before you change this variable!
# default = no
inside_C_lan=no

# do you want to allow CUPS broadcasts?
# this variable works only if  inside_C_lan=yes and is mainly thought
# for tests purposes.
# default = no
allow_cups_b=no

# *** def_ext_int  = default external interface (for the firewall's first start)
# ippp0 - ipppn for ISDN, ppp0 for T-DSL/ADSL or Modems
def_ext_int=ppp0

# *** does this firewall run on a router? (this means that forwarding and
# masquarading is necessary)
# default = yes, otherwise set it to no.
forwarding=yes

# *** the variable "adsl_router" is only necessary if you use this firewall
# on a router with a adsl/t-dsl connection.
# If you do not use this host as a adsl router, set "adsl_router" to no.
adsl_router=yes

# *** Local IP address of this host where this firewall is running
my_host1=192.168.70.9
my_host2=192.168.71.9

# ***  lan1 & lan2  (local area network) are for your regular client-hosts,
# adapt it to your own needs
# if you do not have a lan at all, then comment out (#) lan1 & lan2.
lan1=192.168.70.0/24

# *** if you do not have a second subnet, just remove "lan2" here or
# comment it out.
lan2=192.168.71.0/24

# *** int_if1 (internal interface 1 is for the clients)
int_if1=eth1
int_if2=eth2

# Unprivileged ports
unpriv_p=1024:

#  trace_p are the ports for "traceroute"
trace_p=33434:33523

# *** IP-Address for a transparent Proxy Server (Squid)
# Comment the line "proxy" if you do not have a transparent Proxy Server
#proxy=$my_host

# Listening Port for the Proxy Server
proxy_p=3128
```

```
# Time Server ntp1.ptb.de
timeserver=192.53.103.103

# anything = The Internet
any=0/0

# list of illegal IP addresses
class_a=10.0.0.0/8
class_b=172.16.0.0/12
class_c=192.168.0.0/16
class_d_multicast=224.0.0.0/4
class_e_reserved=240.0.0.0/5
loopback=127.0.0.1/8
broadcast_src=0.0.0.0
broadcast_dst=255.255.255.255
# Determines all the interfaces
# ----------------------
if [ $1 = start ]
then
    # Determines the ISDN or ADSL interface
    if [ $2 ]
    then
      www_if=$2
    else
      www_if=$def_ext_int
    fi

    # Makes a list of all used interfaces
    all_if="$www_if $int_if1 $int_if2"

    # Determines the local IP on the external interface to Internet
    www_ip=$($if_config $www_if |grep "inet [Aa]d" |cut -d: -f 2 \
      |cut -d" " -f 1)

  # determines if a remote connection with SSH for administration purposes
  # is allowed. ssh_rip = ssh remote IP address
    if [ $3 ]
    then
      ssh_rip=$3
    fi
fi
case "$1" in
  start)
    echo

    #------------------------------------------------------------
    echo MyFirewall: Interface=$www_if Local-IP-Address=$www_ip

    # Turning on dynamic kernel parameters
    #------------------------------------------------------------
    echo 1 > /proc/sys/net/ipv4/tcp_syncookies
    echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
    echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

    for f in $all_if; do
       echo 1  > /proc/sys/net/ipv4/conf/$f/rp_filter
       echo 0  > /proc/sys/net/ipv4/conf/$f/accept_redirects
       echo 0  > /proc/sys/net/ipv4/conf/$f/accept_source_route
```

```
      echo 0  > /proc/sys/net/ipv4/conf/$f/bootp_relay
   #  echo 0  > /proc/sys/net/ipv4/conf/$f/log_martians
   done
   echo 1  > /proc/sys/net/ipv4/conf/$www_if/log_martians

   # the following parameters don't exist anymore with SuSE 7.3 and onwards
   file_exists="/proc/sys/net/ipv4/icmp_destunreach_rate"
   test -e $file_exists && echo 5 > $file_exists

   file_exists="/proc/sys/net/ipv4/icmp_echoreply_rate"
   test -e $file_exists && echo 5 > $file_exists

   file_exists="/proc/sys/net/ipv4/icmp_paramprob_rate"
   test -e $file_exists && echo 5 > $file_exists

   file_exists="/proc/sys/net/ipv4/icmp_timeexceed_rate"
   test -e $file_exists && echo 10 > $file_exists
```

```
  # Load the module ip_tables and remove ipchains if allready loaded
   modprobe -r ipchains
   modprobe ip_tables
```

```
  # Set default policies
    echo  "Setting up firewall rules..."
   $iptables -P INPUT   DROP
   $iptables -P OUTPUT  DROP
   $iptables -P FORWARD DROP

   if [ $forwarding = "yes" ]
   then
      $iptables -t nat -P PREROUTING  DROP
      $iptables -t nat -P POSTROUTING DROP
      $iptables -t nat -P OUTPUT      DROP
   fi

   # Flushes all rules of all policies + nat table
   $iptables -F
   $iptables -t nat -F
   ######################Custom chains#############################
   $iptables -N logdropspoof
   $iptables -N logdropopen
   $iptables -N tcp_flags
   $iptables -N icmp_in
   $iptables -N icmp_out
   $iptables -N spoofed_src_ip
   $iptables -N spoofed_dst_ip
   $iptables -N com_check
   $iptables -N www_serv
```

***** logdropspoof** chain *******

(log & drop spoofed packages)

```
   $iptables -A logdropspoof -j LOG --log-prefix "spoofed-ip "
   $iptables -A logdropspoof -j DROP
```

***** logdropopen** chain *******

(log & drop new connections)

```
$iptables -A logdropopen -j LOG --log-prefix "new-or-open "
$iptables -A logdropopen -j DROP
```

*** **tcp_flags** chain ***

(check the validity of tcp flags)

```
# 1st field  = which flags are checked
# 2nd field = which flags are set
$iptables -A tcp_flags -p tcp --tcp-flags ALL NONE        -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags FIN,RST FIN,RST -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags ACK,FIN FIN     -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags ACK,PSH PSH     -j DROP
$iptables -A tcp_flags -p tcp --tcp-flags ACK,URG URG     -j DROP
```

***** icmp_in** chain *******

(accepts some icmp types)

```
$iptables -A icmp_in -p icmp --fragment -j LOG --log-prefix \
                                                  "fragmented "
$iptables -A icmp_in -p icmp --fragment               -j DROP
$iptables -A icmp_in -p icmp --icmp-type echo-reply   -j ACCEPT

$iptables -A icmp_in -p icmp --icmp-type echo-request  \
                             -m limit --limit 5/minute -j ACCEPT
$iptables -A icmp_in -p icmp --icmp-type echo-request  -j DROP

$iptables -A icmp_in -p icmp --icmp-type destination-unreachable \
                                                      -j ACCEPT
$iptables -A icmp_in -p icmp --icmp-type source-quench -j ACCEPT
$iptables -A icmp_in -p icmp --icmp-type time-exceeded -j ACCEPT
$iptables -A icmp_in -p icmp --icmp-type parameter-problem \
                                                      -j ACCEPT
```

***** icmp_out** chain ********

(accepts some icmp types)

```
# accept some ICMP
$iptables -A icmp_out -p icmp --icmp-type echo-reply   -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type echo-request -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type destination-unreachable \
                                                       -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type fragmentation-needed   \
                                                       -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type source-quench          \
                                                       -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type time-exceeded          \
                                                       -j ACCEPT

$iptables -A icmp_out -p icmp --icmp-type parameter-problem      \
                                                       -j ACCEPT
```

*** **spoofed_src_ip** chain ***

(list of invalid source IP addresses)

```
# The following lines are taken from www.linux-firewall-tools/linux
# Refuse addresses defined as reserved by the IANA
# IANA = Internet Assigned Numbers Authority (www.iana.org)
# Note:  this list includes the loopback, multicast,
# and reserved addresses.
# 0.*.*.*- Can't be blocked for DHCP users.

$iptables -A spoofed_src_ip -s $class_a           -j logdropspoof
$iptables -A spoofed_src_ip -s $class_b           -j logdropspoof
    if [ $inside_C_lan = "no" ]
    then
      $iptables -A spoofed_src_ip -s $class_c     -j logdropspoof
    fi
$iptables -A spoofed_src_ip -s $class_d_multicast -j logdropspoof
$iptables -A spoofed_src_ip -s $class_e_reserved  -j logdropspoof
$iptables -A spoofed_src_ip -s $loopback          -j logdropspoof
$iptables -A spoofed_src_ip -s 0.0.0.0/8          -j logdropspoof
$iptables -A spoofed_src_ip -s 169.254.0.0/16     -j logdropspoof
$iptables -A spoofed_src_ip -s 192.0.2.0/24       -j logdropspoof
$iptables -A spoofed_src_ip -s $broadcast_src     -j logdropspoof
```

***** spoofed_dst_ip** chain *******

(list of invalid destination IP addresses)

```
$iptables -A spoofed_dst_ip -d $broadcast_dst     -j logdropspoof
$iptables -A spoofed_dst_ip -p ! udp -d $class_d_multicast \
                                                  -j logdropspoof
```

***** com_check** chain *******

(common check to INPUT & FORWARD chains)

```
# Checks tcp flags
$iptables -A com_check -p tcp -j tcp_flags

# Addresse Spoofing: no packets comming in schould claim to be from lan 1 or 2
    if [ $lan1 ]
    then
      $iptables -A com_check -i $www_if -s $lan1 -j logdropspoof
    fi
    if [ $lan2 ]
    then
      $iptables -A com_check -i $www_if -s $lan2 -j logdropspoof
    fi

    $iptables -A com_check -i $www_if -j spoofed_src_ip
    $iptables -A com_check -i $www_if -j spoofed_dst_ip

     # reject all UDP connections started from the Internet
     # on listening port >= 1024, (for eg. NFS)
     # but except port related to DNS
    for udp_p in $($net_stat -nlpu | grep -v named | \
                  cut -d: -f2 | cut -d" " -f1 | \
```

```
                  sed -n '/[0-9].*/p'); do
      if [ $udp_p -ge 1024 ]; then
         $iptables -A com_check -p udp -i $www_if --dport $udp_p \
                                                  -j logdropopen
      fi
    done

# reject anything + log to X Window ports
$iptables -A com_check -p tcp  -i $www_if --dport 6000:6063 \
                                                  -j logdropopen

# reject + log anything to Open Window port
$iptables -A com_check -p tcp  -i $www_if --dport 2000 \
                                                  -j logdropopen
# reject + log anything NFS & RPC port
# udp ports are already taken care above
$iptables -A com_check -p tcp  -i $www_if --dport 2049 \
                                                  -j logdropopen
$iptables -A com_check -p tcp  -i $www_if --dport 111  \
                                                  -j logdropopen
```

*** www_serv Chain***

(output and forward to the Internet Services)

```
================================================================
```

```
# DNS
$iptables -A www_serv -p tcp --sport $unpriv_p --dport domain    \
                                                    -j ACCEPT

$iptables -A www_serv -p udp --sport $unpriv_p --dport domain    \
                                                    -j ACCEPT

----------------------------------------------------------------
# HTTP & HTTPS (WWW)
$iptables -A www_serv -p tcp --sport $unpriv_p --dport http      \
                                                    -j ACCEPT

$iptables -A www_serv -p tcp --sport $unpriv_p --dport https     \
                                                    -j ACCEPT

----------------------------------------------------------------
# IMAP, POP3 & SMTP (Mail)
$iptables -A www_serv -p tcp --sport $unpriv_p --dport imap      \
                                                    -j ACCEPT

$iptables -A www_serv -p tcp --sport $unpriv_p --dport pop3      \
                                                    -j ACCEPT

$iptables -A www_serv -p tcp --sport $unpriv_p --dport smtp      \
                                                    -j ACCEPT

----------------------------------------------------------------
# FTP (outgoing, control port)
$iptables -A www_serv -p tcp --sport $unpriv_p --dport ftp       \
                                                    -j ACCEPT

----------------------------------------------------------------
# FTP DATA (outgoing, passive data connection)
```

```
$iptables -A www_serv -p tcp --sport $unpriv_p --dport $unpriv_p \
                                                    -j ACCEPT
-------------------------------------------------------------------
# SSH
$iptables -A www_serv -p tcp --sport $unpriv_p --dport ssh       \
                                                    -j ACCEPT
-------------------------------------------------------------------
# Traceroute
$iptables -A www_serv -p udp --sport $unpriv_p --dport $trace_p  \
                                                    -j ACCEPT
-------------------------------------------------------------------
# Auth
$iptables -A www_serv -p tcp --sport $unpriv_p --dport auth      \
                                                    -j ACCEPT
-------------------------------------------------------------------
# Time Server
$iptables -A www_serv -p udp --sport ntp -d $timeserver          \
                                        --dport ntp    -j ACCEPT
```

*** INPUT chain ***

```
# accept everything comming from loopback device
$iptables -A INPUT -i lo -j ACCEPT

# accept everything comming from the lan 1 & 2
if [ $lan1 ]
    then
      $iptables -A INPUT -i $int_if1 -s $lan1 -j ACCEPT
fi
if [ $lan2 ]
    then
      $iptables -A INPUT -i $int_if2 -s $lan2 -j ACCEPT
fi

# accepts some icmp_types
$iptables -A INPUT -i $www_if -p icmp -j icmp_in

# checks TCP Flags , logand drop all possible known spoofed adresses
$iptables -A INPUT -i $www_if -j com_check
-------------------------------------------------------------------
# Transparent Proxy for all clients
    if [ $proxy ]
    then
      $iptables -t nat -A PREROUTING -i $int_if1 -p tcp  \
                --sport $unpriv_p -d ! $proxy --dport 80 \
                -j REDIRECT --to-port $proxy_p
    fi
-------------------------------------------------------------------
```

#Allow access to an **SSH server**?.

```
# Administration entry: ssh only possible with a defined IP address or "ena"
  if [ $ssh_rip != "dis" ]
    then
      if [ $ssh_rip = "ena" ]
      then
          $iptables -A INPUT -p tcp -i $www_if -s $any \
                        --sport $unpriv_p             \
                        -d $www_ip --dport ssh        \
                        -m state --state NEW -j ACCEPT
      else
      $iptables -A INPUT -p tcp -i $www_if -s $ssh_rip \
                        --sport $unpriv_p             \
                        -d $www_ip --dport ssh        \
                        -m state --state NEW -j ACCEPT
      fi
    fi
-----------------------------------------------------------------
# Allow access to a mail server?
    if [ $allow_smtp = "yes" ]
    then
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
                        --sport $unpriv_p             \
                        -d $www_ip --dport smtp       \
                        -m state --state NEW -j ACCEPT
    fi
-----------------------------------------------------------------
# Allow access to a POP-3 server?
    if [ $allow_pop3 = "yes" ]
    then
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
                        --sport $unpriv_p             \
                        -d $www_ip --dport pop3       \
                        -m state --state NEW -j ACCEPT
    fi

-----------------------------------------------------------------
# Allow access to a FTP server?
    if [ $allow_ftp = "yes" ]
    then
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
                        --sport $unpriv_p             \
                        -d $www_ip --dport ftp        \
                        -m state --state NEW -j ACCEPT
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
                        --sport $unpriv_p             \
                        -d $www_ip --dport $unpriv_p  \
                        -m state --state NEW -j ACCEPT

    fi
-----------------------------------------------------------------
# Allow access to Web server?
    if [ $allow_http = "yes" ]
    then
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
                        --sport $unpriv_p             \
                        -d $www_ip --dport http       \
                        -m state --state NEW -j ACCEPT
      $iptables -A INPUT -p tcp -i $www_if -s $any     \
```

```
                          --sport $unpriv_p             \
                          -d $www_ip --dport https      \
                          -m state --state NEW -j ACCEPT
    fi

-----------------------------------------------------------------
# Are we firewalling a host in a C Lan?
    if [ $inside_C_lan = "yes" ]
    then
      # Allow CUPS broadcasts?
      if [ $allow_cups_b = "yes" ]
      lan_b=$(echo $my_host |cut -d. -f 1-3).255
      then
        $iptables -A INPUT -p udp -i $www_if  --sport 631 \
                          -d $lan_b --dport 631 -j ACCEPT
      fi
    fi

-----------------------------------------------------------------
# accept replies only when the connections has been started by oneself
$iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

-----------------------------------------------------------------
# Drop without logging: NetBios,Windows,CUPS,DHCP
$iptables -A INPUT -i $int_if1 -p udp --dport 137:138     -j DROP
$iptables -A INPUT -i $int_if1 -p udp --dport 631         -j DROP
$iptables -A INPUT -i $int_if1 -p udp --sport 60002:60004 -j DROP
$iptables -A INPUT -i $int_if1 -p udp -d 255.255.255.255  -j DROP

if [ $lan2 ]
    then
 $iptables -A INPUT -i $int_if2 -p udp --dport 137:138     -j DROP
 $iptables -A INPUT -i $int_if2 -p udp --dport 631         -j DROP
 $iptables -A INPUT -i $int_if2 -p udp --sport 60002:60004 -j DROP
 $iptables -A INPUT -i $int_if2 -p udp -d 255.255.255.255  -j DROP
fi
-----------------------------------------------------------------
#Log and drop New Connections attempt from Internet
$iptables -A INPUT -m state --state NEW,INVALID \
                                   -j LOG --log-prefix "in-new "

$iptables -A INPUT -m state --state NEW,INVALID -j DROP

-----------------------------------------------------------------
# log all surviving incomming packages
$iptables -A INPUT -j LOG --log-prefix "end-in "
```

*** OUTPUT chain ***

```
 ================================================================
```

```
# accept loopback, lan 1 & 2
$iptables -A OUTPUT -o lo -j ACCEPT

    if [ $lan1 ]
    then
      $iptables -A OUTPUT -o $int_if1 -s $my_host1 -d $lan1      \
                                                      -j ACCEPT
    fi
```

```
    if [ $lan2 ]
    then
      $iptables -A OUTPUT -o $int_if2 -s $my_host2 -d $lan2      \
                                                        -j ACCEPT
    fi
```

Checks **TCP flags** integrity
```
$iptables -A OUTPUT -p tcp  -j tcp_flags
```

Allow **some ICMP types**
```
$iptables -A OUTPUT -p icmp -j icmp_out
```

Doesn't allow **illegal destination** IP addresses
```
$iptables -A OUTPUT -j spoofed_dst_ip
```

just for internal **mail server tests**
```
    if [ $allow_smtp_test = "yes" ]
    then
      $iptables -A OUTPUT -p tcp -s $www_ip --sport smtp \
                          -d $any --dport $unpriv_p -j ACCEPT

      $iptables -A OUTPUT -p tcp -s $www_ip --sport smtp \
                          -d $any --dport smtp      -j ACCEPT
    fi
```

---

outgoing **established & related** connections
```
$iptables -A OUTPUT -o $www_if -m state \
                          --state ESTABLISHED,RELATED   -j ACCEPT

$iptables -A OUTPUT -o $www_if -s $www_ip -m state \
                          --state NEW                   -j www_serv

$iptables -A OUTPUT -o $www_if -m state \
                          --state NEW,INVALID           -j DROP
```

```
# drop netbios packets without logging
$iptables -A OUTPUT -p udp --sport 137:138 -j DROP
$iptables -A OUTPUT -p tcp --sport 139     -j DROP

-----------------------------------------------------------------
# log all surviving outgoing packets
$iptables -A OUTPUT -j LOG --log-prefix "end-out "
```

*** NAT / MASQUERADING ***

```
 ================================================================
if [ $forwarding = "yes" ]
    then
-----------------------------------------------------------------
     # Masquerade lan1 ---> Internet
     if [ $lan1 ]
      then
        $iptables -t nat -A POSTROUTING -o $www_if -s $lan1      \
                                                   -j MASQUERADE
     fi
-----------------------------------------------------------------
     # Masquerade lan2 ---> Internet
     if [ $lan2 ]
      then
        $iptables -t nat -A POSTROUTING -o $www_if -s $lan2      \
                                                   -j MASQUERADE
     fi
-----------------------------------------------------------------
     # accept anything going to and from  local loopback (lo)
     $iptables -t nat -A OUTPUT -o lo -j ACCEPT

     #--------------------------
     $iptables -t nat -A OUTPUT -o $www_if  -s $www_ip  -j ACCEPT

     #--------------------------
     if [ $lan1 ]
      then
        $iptables -t nat -A OUTPUT -o $int_if1 \
                                   -s $my_host1 -d $lan1 -j ACCEPT
     fi

     #--------------------------
     if [ $lan2 ]
      then
        $iptables -t nat -A OUTPUT -o $int_if2 \
                                   -s $my_host2 -d $lan2 -j ACCEPT
     fi

     #--------------------------
     $iptables -t nat -A POSTROUTING -o lo                    -j ACCEPT
     $iptables -t nat -A POSTROUTING -o $www_if  -s $www_ip   -j ACCEPT
     $iptables -t nat -A POSTROUTING -o $int_if1 -s $my_host1 -j ACCEPT
     $iptables -t nat -A POSTROUTING -o $int_if2 -s $my_host2 -j ACCEPT
     $iptables -t nat -A POSTROUTING -o $int_if1 -s $lan2     -j ACCEPT
     $iptables -t nat -A POSTROUTING -o $int_if2 -s $lan1     -j ACCEPT

     #--------------------------
     if [ $lan1 ]
      then
        $iptables -t nat -A PREROUTING  -i $int_if1 \
                                           -s $lan1 -j ACCEPT
     fi
```

```
     #-------------------------
     if [ $lan2 ]
      then
        $iptables -t nat -A PREROUTING  -i $int_if2 \
                                            -s $lan2 -j ACCEPT
     fi
```

```
     # Allow access to a mail server?
     if [ $allow_smtp = "yes" ]
      then
        $iptables -t nat -A PREROUTING -p tcp -i $www_if    \
                         -s $any --sport smtp               \
                         -d $www_ip --dport smtp -j ACCEPT

        $iptables -t nat -A PREROUTING -p tcp -i $www_if    \
                         -s $any --sport $unpriv_p          \
                         -d $www_ip --dport smtp -j ACCEPT
      fi
```

```
     # Allow SSH?
      if [ $ssh_rip != "dis" ]
      then
        if [ $ssh_rip = "ena" ]
        then
          $iptables -t nat -A PREROUTING -p tcp -i $www_if  \
                           -s $any --sport $unpriv_p        \
                           -d $www_ip --dport ssh -j ACCEPT

          $iptables -t nat -A PREROUTING -p tcp -i $www_if  \
                           -s $any --sport $unpriv_p        \
                           -d $www_ip --dport $unpriv_p -j ACCEPT
        else
          $iptables -t nat -A PREROUTING -p tcp -i $www_if  \
                           -s $ssh_rip --sport $unpriv_p    \
                           -d $www_ip  --dport ssh -j ACCEPT

          $iptables -t nat -A PREROUTING -p tcp -i $www_if  \
                           -s $ssh_rip --sport $unpriv_p    \
                           -d $www_ip  --dport $unpriv_p -j ACCEPT
        fi
      fi
```

```
    #Accept echo requests ICMP packets
    $iptables -t nat -A PREROUTING -p icmp --icmp-type      \
                                      echo-request -j ACCEPT
```

```
    #Drop the Netbios packets without logging
    $iptables -t nat -A OUTPUT        -p udp --sport 137:138 -j DROP
    $iptables -t nat -A OUTPUT        -p udp --dport 137:138 -j DROP
    $iptables -t nat -A PREROUTING    -p udp --sport 137:138 -j DROP
    $iptables -t nat -A PREROUTING    -p udp --dport 137:138 -j DROP
```

```
    # Log the surviving packets
    $iptables -t nat -A PREROUTING  -j LOG --log-prefix "end-nat-pre "
```

```
    $iptables -t nat -A POSTROUTING -j LOG --log-prefix "end-nat-post "
    $iptables -t nat -A OUTPUT      -j LOG --log-prefix "end-nat-out "
```

*** FORWARD chain ***

==========

turn on **IP Forwarding** and **Dynamic Address**

```
   echo 1 > /proc/sys/net/ipv4/ip_forward
   echo 1 > /proc/sys/net/ipv4/ip_dynaddr
```

---

correction of **MSS** (Max Segment Size) for **ADSL clients**
1452 Bytes + 40 Bytes (TCP Header) + 8 Bytes (PPPoE) = 1500 Bytes

```
   if [ $adsl_router = "yes" ]
   then
     $iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN \
                          -j TCPMSS --clamp-mss-to-pmtu
   fi
```

```
   #Forward related Established Connection responses from outside
   $iptables -A FORWARD -i $www_if  -o $int_if1 \
                   -m state --state ESTABLISHED,RELATED -j ACCEPT

   $iptables -A FORWARD -i $int_if1 -o $www_if  \
                   -m state --state ESTABLISHED,RELATED -j ACCEPT

   if [ $lan2 ]
      then
          $iptables -A FORWARD -i $www_if  -o $int_if2 \
                   -m state --state ESTABLISHED,RELATED -j ACCEPT

          $iptables -A FORWARD -i $int_if2 -o $www_if  \
                   -m state --state ESTABLISHED,RELATED -j ACCEPT
   fi

-----------------------------------------------------------------
   # Log and drop all New connections from Internet
   $iptables -A FORWARD -i $www_if  -o $int_if1 \
                           -m state --state NEW,INVALID \
                           -j LOG --log-prefix "forw-drop "

   $iptables -A FORWARD -i $www_if  -o $int_if1 \
                           -m state --state NEW,INVALID -j DROP
   if [ $lan2 ]
      then
          $iptables -A FORWARD -i $www_if  -o $int_if2 \
                           -m state --state NEW,INVALID \
                           -j LOG --log-prefix "forw-drop "

          $iptables -A FORWARD -i $www_if  -o $int_if2 \
                           -m state --state NEW,INVALID -j DROP
   fi

-----------------------------------------------------------------
   # allow some ICMP types
   $iptables -A FORWARD -p icmp -j icmp_out

-----------------------------------------------------------------
   # checks TCP Flags and spoofed IPs
   $iptables -A FORWARD -i $www_if  -j com_check

-----------------------------------------------------------------
```

```
    # Forward services initiated from lans: lan1&2 ----> Internet
    if [ $lan1 ]
     then
       $iptables -A FORWARD -i $int_if1 -s $lan1 -o $www_if \
                            -m state --state NEW -j www_serv
    fi

    if [ $lan2 ]
       then
         $iptables -A FORWARD -i $int_if2 -s $lan2 -o $www_if \
                            -m state --state NEW -j www_serv
    fi

-------------------------------------------------------------------
    #  Forward all packets from Lan1 to Lan2: lan1 ----> lan2
    if [ $lan1 -a $lan2 ]
       then
         $iptables -A FORWARD -i $int_if1 -s $lan1 -o $int_if2    \
                                          -d $lan2 -j ACCEPT

         $iptables -A FORWARD -i $int_if2 -s $lan2 -o $int_if1    \
                                          -d $lan1 -j ACCEPT
       fi

     # Don't show the rejected UDP-Packets
     $iptables -A FORWARD -p udp --sport 1024: -j DROP

-------------------------------------------------------------------
     # catch all surviving packets for logging
     $iptables -A FORWARD -j LOG --log-prefix "end-forw "

---------- END OF ROUTER RULES (forwarding = yes)----------
=========================================================

else
      # Turn OFF forwarding and dynamic adressing
       echo 0 > /proc/sys/net/ipv4/ip_forward
       echo 0 > /proc/sys/net/ipv4/ip_dynaddr
fi

     echo -e "$rc_done"

;;
```

#---------- **STOP FIREWALL** ----------

```
  stop)
    echo -n "shutting down firewall rules. "

    # Turning off IP Forwarding
    echo 0 > /proc/sys/net/ipv4/ip_forward

    # Set default policies
    $iptables -P INPUT  ACCEPT
    $iptables -P OUTPUT ACCEPT
    $iptables -P FORWARD DROP

    $iptables -t nat -P PREROUTING  ACCEPT
    $iptables -t nat -P POSTROUTING ACCEPT
    $iptables -t nat -P OUTPUT      ACCEPT

    # Flushing(clearing) all rules of all tables
    $iptables -F
    $iptables -t nat -F

    # Delete customized chains
    $iptables -X

    echo -e "$rc_done"
    ;;

  restart)
    $0 stop && $0 start $2 $3 || echo -e " $rc_failed"
    ;;

  status)
    # Show all rules of all tables
    $iptables -nvL
    echo " "
    echo "--- *** NAT-TABLE ***------------------------------"
    echo " "
    $iptables -t nat -nvL
    ;;

  *)
    # Display error if arguments syntax is incorrect
    echo -n "Usage: $0 {start|stop|restart|status}"
    echo -e "$rc_failed"
    exit 1

esac

exit 0
```