MyFirewall (Pierre Burri)

```
#!/bin/bash
# Copyright (c) 2002-2003 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   : 1.2r09
# Author    : Pierre Burri, can be reached at p.burri@linux-age.com
#             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.
# Licence   : GNU GENERAL PUBLIC LICENCE (GPL)
# 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 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 ifconfig,
#                      netstat 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
#           16-Oct-2002  - corrected a spelling mistake (int_inf1 -> int_if1)
#           17-Oct-2002  - added echo-reply in icmp_out
#           28-Oct-2002  - added protocols for VPN-Clients.
#           29-Oct-2002  - added variable for switching logging on/off
#           13-Nov-2002  - allow pop3s - "pop3 over ssl" (Port 995)
#           07-Dec-2002  - allow imaps - "imap over ssl" (Port 993)
#           30-Mar-2003  - allow a second ntp server
#                    - read variable from /root/.MyFirewall.config
#                      if file exists.
#           31-Mar-2003  - remove filters on nat tables
#            5-Apr-2003  - read variable from this file in any case
#                        and read afterward the variable from
#                        .MyFirewall.config
#                    - possibility to close some unprivilege ports
#                      if ftp is open (unpriv_ports_to_close)
#                    - added allow_webmin
#                    - dmz_server1 added for test purposes
#           16-Apr-2003  - added the following variables:
#                        allow_ipsec, bypass_www_serv_input, nb_pings,
#                        bypass_www_serv_forward, allow_www_ip_as_src_for_lan,
#                        forward_ports_to_close, allow_hylafax, hylafax_ports
#                            - renamed unpriv_ports_to_close to
#                              input_ports_to_close
#             23-Apr-2003  - added variable allow_only_first_parameter
#                        and a warning message comes on the terminal.
#             25-Apr-2003  - added variables udp_ports_to_not_log &
#                        tcp_ports_to_not_log
#                    - added variable forward_subnet_to_close
#            1-May-2003  - added rules and variable for ntpd
#             26-May-2003  - bug in the rules sequence in FORWARD fixed.
#           27-May-2003  - bug in forward_ports_to_close fixed
#                        (www_if instead of $www_if).
#
# 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.
#
#           Since version 1.2, MyFirewall reads its variables from this file
#           and afterward from file /root/.MyFirewall.config if it exists.
#             This means that the variables defined in .MyFirewall.config will
#           overwrite the variable defined here!
#           .MyFirewall.config should have a file access of 640 or 600.
#
#             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
#             A list of DNS Servers can be found on the following web page:
#             http://www.fli4l.de/german/dns.htm
#           (the following example is for DNS servers of T-Online):
#
#           forwarders { 194.25.2.129; 194.25.2.130; };
```

```
#
#             and configure carefully who can access your name server, eg.:
#
#           allow-query { 127.0.0.1; 192.168.30.0/24; };
#
#================================================================================
if [ $TERM = "linux" -o $TERM = "xterm" ]
then
  col_70="\015\033[80C\033[10D"
  green_on="\033[1;32m"
  red_on="\033[1;31m"
  color_off="\033[m\017"
  rc_done=$col_70$green_on"done"$color_off
  rc_failed=$col_70$red_on"failed"$color_off
else
  rc_done="done"
  rc_failed="failed"
fi

# Variables with 3 stars *** have to be adapted!
# ==============================================
#
# All or some of these variables will be overwritten if you have the personal
# configuration file /root/.MyFirewall.config !!!
#
# 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
# For Debian and Red Hat: iptables=/sbin/iptables
ifconfig="/sbin/ifconfig"
netstat="/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="dis"

# *** 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 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"
allow_pop3s="no"
allow_imaps="no"

# *** do you want to allow access to your FTP server from outside?
# default = no
allow_ftp="no"

# if allow_ftp=yes, all unprivilege ports will be open. input_ports_to_close
# allows to close all ports that we don't want to be suddenly open.
# input_ports_to_close can be a list of several ports.
# input_ports_to_close="8081 10000"
input_ports_to_close=""

# similar to input_ports_to_close, it is possible to block some ports that
# are normally forwarded, undepedently of the variable allow_ftp.
forward_ports_to_close=""

# Since this firewall is also used in certains schools, it might be
# desirable to stop some file sharing servers like eDonkey, Gnutella,
# Napster, Morpheus etc. Some students otherwise would use almost the
# whole bandwidth for themselves...
# Both variables forward_ports_to_close & forward_subnet_to_close
# can be used for this.
#-----------------------------------------------------------------
# Aimster:   TCP 5025
# Bearshare       TCP 6346
# DirectConnect: TCP 411 412
# eDonkey:   TCP 4661 4662 UDP 4665
# Gnutella:  TCP 6346
# Grokster:  TCP 1214
# Hotline:   TCP 1234 5498 5499 5500 5501
# KaZaA:     TCP 1214
# LimeWire:  TCP 6346 6347
# Napster:   TCP 6699
# ToadNode:  TCP 6346
# WinMX:     TCP 6699 UDP 6257
# Xolox:     TCP 6346
#-----------------------------------------------------------------
# Audiogalaxy:     64.245.58.0/24  64.245.59.0/24
# IMesh:     216.35.208.0/24
# Napigator:       209.25.178.0/24
# Napster:   64.124.41.0/24
# WinMX:     209.61.186.0/24 64.49.201.0/24
#-----------------------------------------------------------------
forward_subnet_to_close=""

# do you want to allow access to your hylafax server or hylafax ftp server?
allow_hylafax="no"
# The old port is 4557, the new one is 4559
# hylafax_ports="4557 4559"
hylafax_ports="4559"
```

```
# do you want to allow access to your webmin interface?
# webmin should be setup with SSL!
# default = no
allow_webmin=no
webmin_port="10000"

# allow redirection to web, mail, ftp servers in a DMZ
# dmz_server1=192.168.30.250
# dmz_server1_ports="80 21"

# 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"

# do you want to allow VPN-Clients?
# Prococol esp (50) and UDP port 500.
# default = no
allow_vpn_clients="no"
# put here the IP-Address of your VPN-Servers
vpn_server1="0/0"
# if do not have a second VPN-Server, just remove it.
#vpn_server2=0/0

# do you allow full ipsec traffic?
allow_ipsec="no"

# do you want to log dropped packets?
# the logfiles are in /var/log/messages and by SuSE also in
# /var/log/firewall.
# default for log_on = yes. default for log_nat_on = no
log_on="yes"
log_nat_on="no"

# some ports create a lot of entries in the log file. the two following
# variables allow to drop these port so there are not logged.
udp_ports_to_not_log="137 138"
tcp_ports_to_not_log="139"

# *** 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"

# *** int_if1 (internal interface 1 is for the clients)
int_if1="eth0"
# int_if2 (internal interface 2 can be for a DMZ)
#int_if2="eth2"

# first ipsec interface
ipsec_if0="ipsec0"

# *** IP address of this host where this firewall is running
my_host=$($ifconfig $int_if1 |grep "inet [Aa]d" |cut -d: -f 2 |cut -d" " -f 1)

# *** lan 1 & 2 (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="192.168.30.0/24"

# *** if you do not have a second subnet, just remove "lan2" here or
# comment it out. "lan2" an be used for example as a DMZ.
#lan2="192.168.50.0/24"

# unprivileged ports
unpriv_p="1024:"

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

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

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

# Time Servers
# For a NTP-Server List see: http://www.eecis.udel.edu/~mills/ntp/clock1a.html
# ntp3.fau.de
ntp_server1="131.188.3.223"
# ntps1-1.cs.tu-berlin.de
#ntp_server2="130.149.17.8"
# The ntp daemon (ntpd/xntpd) uses one of the unpriv sports
# The client program ntpdate uses sport ntp (123)
allow_ntpd="no"

# allow to bypass rules of chain www_serv
bypass_www_serv_output="no"
bypass_www_serv_forward="no"

# as to be set in certain special cases
# e.g. ping to the own internet IP address
allow_www_ip_as_src_for_lan="no"

# defines how many pings are allowed per minutes
# default is 5 per minutes
nb_pings="5"
```

```
# allows to disable the entrance of parameter $2 (external interface)
# and $3 (ssh remote access) on the command line. This is very usefull
# for servers that are administrated remotely, to avoid to block the
# own access over ssh.
# default value = no
allow_only_first_parameter="no"

# 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="127.0.0.0/8"
broadcast_src="0.0.0.0"
broadcast_dst="255.255.255.255"

# Takes variable definition from /root/.MyFirewall.config
if [ -e /root/.MyFirewall.config ]
then
  . /root/.MyFirewall.config
fi

# Determines all the interfaces
# -----------------------------
if [ $1 = start ]
then
   # Determines the ISDN or ADSL interface
   if [ $2 ]
   then
     if [ $allow_only_first_parameter = "no" ]
     then
       www_if=$2
     else
       if [ $TERM = "linux" -o $TERM = "xterm" ]
       then
          echo -e $red_on"Only first parameter \
                  (start|stop|restart|status) is allowed!"$color_off
       fi
       www_if=$def_ext_int
     fi
   else
     www_if=$def_ext_int
   fi

   # Makes a list of all used interfaces
   all_if="$www_if $int_if1"
   if [ $int_if2 ]
   then
     all_if="$www_if $int_if1 $int_if2"
   fi

   # Determines the local IP on the external interface
   www_ip=$($ifconfig $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
     if [ $allow_only_first_parameter = "no" ]
     then
       ssh_rip=$3
     fi
   fi
fi
#--------------------------------
START-------------------------------------

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

    # the following parameters don't exist anymore with SuSE 7.3
    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

    for f in $all_if; do
       if [ $allow_ipsec = "no" ]
       then
         echo 1  > /proc/sys/net/ipv4/conf/$f/rp_filter
       else
         echo 0  > /proc/sys/net/ipv4/conf/$f/rp_filter
       fi
       echo 0  > /proc/sys/net/ipv4/conf/$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 1  > /proc/sys/net/ipv4/conf/$f/log_martians
    done

    # 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

    # Flushes all rules of all policies + nat table
    $iptables -F
    $iptables -t nat -F

    # Create customized chains
    $iptables -N www_serv
```

```
$iptables -N com_check
$iptables -N spoofed_src_ip
$iptables -N spoofed_dst_ip
$iptables -N icmp_in
$iptables -N icmp_out
$iptables -N tcp_flags
$iptables -N logdropspoof
$iptables -N logdropopen

# *** logdropspoof chain *** = log & drop spoofed packages
# ------------------------------------------------------------------------
if [ $log_on = "yes" ]
then
  $iptables -A logdropspoof -j LOG --log-prefix "spoofed-ip "
fi
$iptables -A logdropspoof -j DROP

# *** logdropopen chain *** = log & drop new connections
# ------------------------------------------------------------------------
if [ $log_on = "yes" ]
then
  $iptables -A logdropopen -j LOG --log-prefix "new-or-open "
fi
$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
# ------------------------------------------------------------------------
if [ $log_on = "yes" ]
then
   $iptables -A icmp_in -p icmp --fragment -j LOG --log-prefix "fragmented
"
fi
$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 $nb_pings/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
```

```
# *** 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 spoofed addresses, tcp flags and UDP open/listening ports.
$iptables -A com_check -p tcp -j tcp_flags

# Spoofing Protection:
# 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 $($netstat -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


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

if [ $allow_ipsec = "yes" ]
then
  $iptables -A INPUT -p udp --sport 500 --dport 500 -j ACCEPT
  $iptables -A INPUT -p 50  -j ACCEPT
  $iptables -A INPUT -p 51  -j ACCEPT
fi

# 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
if [ $allow_ipsec = "no" ]
then
  $iptables -A INPUT -i $www_if -p icmp -j icmp_in
else
  $iptables -A INPUT            -p icmp -j icmp_in
fi
# checks TCP Flags and
# log and drop all possible known spoofed adresses
$iptables -A INPUT -i $www_if -j com_check


# Admin 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 "POP-3 over SSL" server?
if [ $allow_pop3s = "yes" ]
then
  $iptables -A INPUT -p tcp -i $www_if -s $any --sport $unpriv_p \
                     -d $www_ip --dport pop3s \
                     -m state --state NEW -j ACCEPT
fi

# Allow access to a "IMAP over SSL" server?
if [ $allow_imaps = "yes" ]
then
  $iptables -A INPUT -p tcp -i $www_if -s $any --sport $unpriv_p \
                     -d $www_ip --dport imaps \
                     -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

  # closes some ports that are suddenly open because of ftp
  # for example 8081 is for tomcat
  if [ "$input_ports_to_close" != "" ]
  then
    for p in $input_ports_to_close
  do
      $iptables -A INPUT -p tcp -i $www_if -s $any --sport $unpriv_p \
                         -d $www_ip --dport $p -j DROP
  done
  fi

  $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 a Hylafax server?
if [ $allow_hylafax = "yes" ]
then
  if [ "$hylafax_ports" != "" ]
  then
    for p in $hylafax_ports
  do
      $iptables -A INPUT -p tcp -i $www_if -s $any --sport $unpriv_p \
                         -d $www_ip --dport $p -j ACCEPT
  done
  fi
fi

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

# Inside_c_lan?
if [ $inside_C_lan = "yes" ]
then
  # Allow CUP 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

# 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

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

# doesn't log certain ports that fill up the log files
if [ "$udp_ports_to_not_log" != "" ]
then
  for p in $udp_ports_to_not_log
  do
      $iptables -A INPUT -p udp --dport $p -j DROP
  done
fi
if [ "$tcp_ports_to_not_log" != "" ]
then
  for p in $tcp_ports_to_not_log
  do
      $iptables -A INPUT -p tcp --dport $p -j DROP
  done
fi

if [ $log_on = "yes" ]
then
  $iptables -A INPUT -m state --state NEW,INVALID \
                              -j LOG --log-prefix "in-new "
```

```
fi

# log all surviving incomming packages
# ------------------------------------------------------------------------
if [ $log_on = "yes" ]
then
  $iptables -A INPUT -j LOG --log-prefix "end-in "
fi

# drops invalid new packages (SYN)
$iptables -A INPUT -m state --state NEW,INVALID -j DROP


# *** icmp_out ****
# ========================================================================
# 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


# *** www_serv *** = 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
$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, IMAPS, POP3, POP3S & SMTP
$iptables -A www_serv -p tcp --sport $unpriv_p --dport imap   -j ACCEPT
$iptables -A www_serv -p tcp --sport $unpriv_p --dport imaps  -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 pop3s  -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 Servers (ntpdate uses sport ntp, xntpdd uses sport 1024:)
$iptables -A www_serv -p udp --sport ntp -d $ntp_server1 --dport ntp \
                                                                 -j ACCEPT
if [ $ntp_server2 ]
then
  $iptables -A www_serv -p udp --sport ntp -d $ntp_server2 --dport ntp \
                                                                 -j ACCEPT
fi
if [ $allow_ntpd = "yes" ]
then
```

```
  $iptables -A www_serv -p udp --sport $unpriv_p -d $ntp_server1 \
                               --dport ntp -j ACCEPT
  if [ $ntp_server2 ]
  then
    $iptables -A www_serv -p udp --sport $unpriv_p -d $ntp_server2 \
                                 --dport ntp -j ACCEPT
  fi
fi

# VPN - Virtual Private Network
if [ $allow_vpn_clients = "yes" ]
then
   $iptables -A www_serv -p esp -d $vpn_server1      -j ACCEPT
   $iptables -A www_serv -p udp --sport 500 \
                         -d $vpn_server1 --dport 500 -j ACCEPT
   if [ $vpn_server2 ]
   then
      $iptables -A www_serv -p esp -d $vpn_server2      -j ACCEPT
      $iptables -A www_serv -p udp --sport 500 \
                            -d $vpn_server2 --dport 500 -j ACCEPT
   fi
fi


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

if [ $allow_ipsec ]
then
  $iptables -A OUTPUT -p udp --sport 500 --dport 500 -j ACCEPT
  $iptables -A OUTPUT -p 50  -j ACCEPT
  $iptables -A OUTPUT -p 51  -j ACCEPT
fi

if [ $lan1 ]
then
  $iptables -A OUTPUT -o $int_if1 -s $my_host -d $lan1  -j ACCEPT
  if [ $allow_www_ip_as_src_for_lan = "yes" ]
  then
    $iptables -A OUTPUT -o $int_if1 -s $www_ip -d $lan1  -j ACCEPT
  fi
fi
if [ $lan2 ]
then
  $iptables -A OUTPUT -o $int_if2 -s $my_host -d $lan2  -j ACCEPT
  if [ $allow_www_ip_as_src_for_lan = "yes" ]
  then
    $iptables -A OUTPUT -o $int_if2 -s $www_ip -d $lan2  -j ACCEPT
  fi
fi

# checks TCP flags integrity and allow some ICMP types
$iptables -A OUTPUT -p tcp  -j tcp_flags
$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
    if [ $bypass_www_serv_output = "yes" ]
    then
      $iptables -A OUTPUT -o $www_if -s $www_ip -m state --state NEW -j ACCEPT
    else
      $iptables -A OUTPUT -o $www_if -s $www_ip -m state --state NEW -j
www_serv
    fi

    # doesn't log certain ports that fill up the log files
    if [ "$udp_ports_to_not_log" != "" ]
    then
      for p in $udp_ports_to_not_log
      do
          $iptables -A OUTPUT -p udp --dport $p -j DROP
      done
    fi
    if [ "$tcp_ports_to_not_log" != "" ]
    then
      for p in $tcp_ports_to_not_log
      do
          $iptables -A OUTPUT -p tcp --dport $p -j DROP
      done
    fi

    # log all surviving outgoing packages
    # ------------------------------------------------------------------------
    if [ $log_on = "yes" ]
    then
      $iptables -A OUTPUT -j LOG --log-prefix "end-out "
    fi

    # drops all invalid new packages (SYN)
    $iptables -A OUTPUT -o $www_if -m state --state NEW,INVALID    -j DROP
```

```
# *** NAT / MASQUERADING  ***
# ========================================================================
if [ $forwarding = "yes" ]
then
  # transparent proxy for all clients
  if [ $proxy ]
  then
    $iptables -t nat -A PREROUTING -i $int_if1 -p tcp  \
              --sport $unpriv_p -d ! $proxy --dport http \
              -j REDIRECT --to-port $proxy_p
  # redirection of https to squid doesn't seem to work!
    #$iptables -t nat -A PREROUTING -i $int_if1 -p tcp  \
    #          --sport $unpriv_p -d ! $proxy --dport https \
    #          -j REDIRECT --to-port $proxy_p
  fi

  # IP Adress Redirection for Servers in a DMZ
  if [ $dmz_server1 ]
  then
    for p in $dmz_server1_ports
  do
      $iptables -t nat -A PREROUTING -i $www_if -p tcp \
            -d $www_ip --dport $p \
            -j DNAT --to-destination $dmz_server1
  done
  fi

  if [ $lan1 ]
  then
    $iptables -t nat -A POSTROUTING -o $www_if -s $lan1 -j MASQUERADE
  fi
  if [ $lan2 ]
  then
    $iptables -t nat -A POSTROUTING -o $www_if -s $lan2 -j MASQUERADE
  fi

  if [ $log_nat_on = "yes" ]
  then
    $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 "
  fi

  # turn on IP Forwarding and dynamic address
  # ========================================================================
  echo 1 > /proc/sys/net/ipv4/ip_forward
  echo 1 > /proc/sys/net/ipv4/ip_dynaddr

  # *** FORWARD chain ***
  # ========================================================================

  if [ $allow_ipsec ]
  then
    if [ $lan1 ]
  then
    $iptables -A FORWARD -i $int_if1   -o $ipsec_if0 -j ACCEPT
    $iptables -A FORWARD -i $ipsec_if0 -o $int_if1   -j ACCEPT
  fi
    if [ $lan2 ]
  then
    $iptables -A FORWARD -i $int_if2   -o $ipsec_if0 -j ACCEPT
    $iptables -A FORWARD -i $ipsec_if0 -o $int_if2   -j ACCEPT
  fi
  fi

  # 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

$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 [ $int_if2 ]
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

# Allow SYN for Servers in a DMZ
if [ $dmz_server1 ]
then
  for p in $dmz_server1_ports
do
    $iptables -A FORWARD -i $www_if -p tcp \
          -d $dmz_server1 --dport $p \
          -m state --state NEW -j ACCEPT
done
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

# closes some ports that are normally open through www_serv
if [ "$forward_ports_to_close" != ""  ]
then
  for p in $forward_ports_to_close
do
  if [ $lan1 ]
  then
      $iptables -A FORWARD -p tcp -i $int_if1 -o $www_if  \
                           -s $lan1 --dport $p -j DROP
      $iptables -A FORWARD -p udp -i $int_if1 -o $www_if  \
                           -s $lan1 --dport $p -j DROP
  fi
  if [ $lan2 ]
  then
      $iptables -A FORWARD -p tcp -i $int_if2 -o $www_if  \
                           -s $lan2 --dport $p -j DROP
      $iptables -A FORWARD -p udp -i $int_if2 -o $www_if  \
                           -s $lan2 --dport $p -j DROP
  fi
done
fi

# closes some subnet that are normally open through www_serv
if [ "$forward_subnet_to_close" != ""  ]
then
  for p in $forward_subnet_to_close
do
  if [ $lan1 ]
  then
      $iptables -A FORWARD -i $int_if1 -o $www_if  \
                           -s $lan1 -d $p -j DROP
  fi
  if [ $lan2 ]
  then
```

```
            $iptables -A FORWARD -i $int_if2 -o $www_if  \
                                 -s $lan2 -d $p -j DROP
        fi
      done
      fi

      if [ $lan1 ]
      then
        if [ $bypass_www_serv_forward = "yes" ]
      then
          $iptables -A FORWARD -i $int_if1 -s $lan1 -o $www_if \
                                       -m state --state NEW -j ACCEPT
      else
          $iptables -A FORWARD -i $int_if1 -s $lan1 -o $www_if \
                                       -m state --state NEW -j www_serv
      fi
      fi
      if [ $lan2 ]
      then
        if [ $bypass_www_serv_forward = "yes" ]
      then
          $iptables -A FORWARD -i $int_if2 -s $lan2 -o $www_if \
                                       -m state --state NEW -j ACCEPT
      else
          $iptables -A FORWARD -i $int_if2 -s $lan2 -o $www_if \
                                       -m state --state NEW -j www_serv
      fi
      fi

      # catch all surviving rules for logging purpose
      # ------------------------------------------------------------------------
      if [ $log_on = "yes" ]
      then
        $iptables -A FORWARD -i $www_if  -o $int_if1 \
                             -m state --state NEW,INVALID \
                             -j LOG --log-prefix "forw-drop "
        if [ $int_if2 ]
      then
          $iptables -A FORWARD -i $www_if  -o $int_if2 \
                               -m state --state NEW,INVALID \
                               -j LOG --log-prefix "forw-drop "
      fi
      fi
      $iptables -A FORWARD -i $www_if  -o $int_if1 \
                       -m state --state NEW,INVALID -j DROP
      if [ $int_if2 ]
      then
        $iptables -A FORWARD -i $www_if  -o $int_if2 \
                           -m state --state NEW,INVALID -j DROP
      fi

      if [ $log_on = "yes" ]
      then
        $iptables -A FORWARD -j LOG --log-prefix "end-forw "
      fi

    # if [ $forwarding = "yes" ]
    else
      echo 0 > /proc/sys/net/ipv4/ip_forward
      echo 0 > /proc/sys/net/ipv4/ip_dynaddr
    fi

    echo -e "$rc_done"
    ;;
#------------------------------- STOP-------------------------------------
  stop)
    echo -n "shutting down firewall rules. "
    # Turning off IP Forwarding
```

```
    echo 0 > /proc/sys/net/ipv4/ip_forward

    # Set default policies & clear all rules
    $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 all rules of all policies
    $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)
    $iptables -nvL
    echo " "
    echo "--- *** NAT-TABLE *** ---------------------------------------------"
    echo " "
    $iptables -t nat -nvL
    ;;

  *)
    echo -n "Usage: $0 {start|stop|restart|status}"
    echo -e "$rc_failed"
    exit 1

esac
exit 0
```