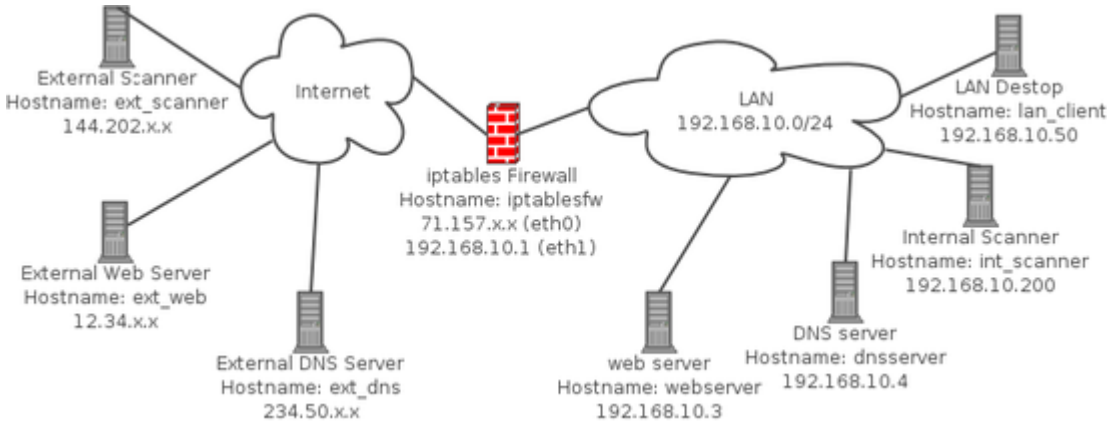
basic iptables policy

Network Topology?



iptables.sh?

```
#!/bin/sh
#
###############################################################################
#
# File: iptables.sh
#
# Purpose: To build a basic iptables policy with default log and drop rules.
# This script was written for the book "Linux Firewalls: Attack
# Detection and Response" published by No Starch Press.
#
# Copyright (C) 2006-2009 Michael Rash (mbr@cipherdyne.org)
#
# License (GNU Public License):
#
# 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
#
#
###############################################################################
#
# updated by alang
#

IPTABLES=/sbin/iptables
```

```
IP6TABLES=/sbin/ip6tables
MODPROBE=/sbin/modprobe
INT_NET=192.168.10.0/24

### flush existing rules and set chain policy setting to DROP
echo "[+] Flushing existing iptables rules..."
$IPTABLES -F
$IPTABLES -F -t nat
$IPTABLES -X
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP

### this policy does not handle IPv6 traffic except to drop it.
#
echo "[+] Disabling IPv6 traffic..."
$IP6TABLES -P INPUT DROP
$IP6TABLES -P OUTPUT DROP
$IP6TABLES -P FORWARD DROP

### load connection-tracking modules
#
$MODPROBE ip_conntrack
$MODPROBE iptable_nat
$MODPROBE ip_conntrack_ftp
$MODPROBE ip_nat_ftp

###### INPUT chain ######
#
echo "[+] Setting up INPUT chain..."

### state tracking rules
$IPTABLES -A INPUT -m state --state INVALID -j LOG --log-prefix "DROP INVALID " --log-ip-options
--log-tcp-options
$IPTABLES -A INPUT -m state --state INVALID -j DROP
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

### anti-spoofing rules
$IPTABLES -A INPUT -i eth1 -s ! $INT_NET -j LOG --log-prefix "SPOOFED PKT "
$IPTABLES -A INPUT -i eth1 -s ! $INT_NET -j DROP

### ACCEPT rules
$IPTABLES -A INPUT -i eth1 -p tcp -s $INT_NET --dport 22 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -j ACCEPT

### default INPUT LOG rule
$IPTABLES -A INPUT -i ! lo -j LOG --log-prefix "DROP " --log-ip-options --log-tcp-options

### make sure that loopback traffic is accepted
$IPTABLES -A INPUT -i lo -j ACCEPT
```

```
###### OUTPUT chain ######
#
echo "[+] Setting up OUTPUT chain..."

### state tracking rules
$IPTABLES -A OUTPUT -m state --state INVALID -j LOG --log-prefix "DROP INVALID " --log-ip-options
--log-tcp-options
$IPTABLES -A OUTPUT -m state --state INVALID -j DROP
$IPTABLES -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

### ACCEPT rules for allowing connections out
$IPTABLES -A OUTPUT -p tcp --dport 21 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 22 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 25 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 43 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 80 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 443 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 4321 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --dport 53 -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p udp --dport 53 -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

### default OUTPUT LOG rule
$IPTABLES -A OUTPUT -o ! lo -j LOG --log-prefix "DROP " --log-ip-options --log-tcp-options

### make sure that loopback traffic is accepted
$IPTABLES -A OUTPUT -o lo -j ACCEPT

###### FORWARD chain ######
#
echo "[+] Setting up FORWARD chain..."

### state tracking rules
$IPTABLES -A FORWARD -m state --state INVALID -j LOG --log-prefix "DROP INVALID " --log-ip-options
--log-tcp-options
$IPTABLES -A FORWARD -m state --state INVALID -j DROP
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

### anti-spoofing rules
$IPTABLES -A FORWARD -i eth1 -s ! $INT_NET -j LOG --log-prefix "SPOOFED PKT "
$IPTABLES -A FORWARD -i eth1 -s ! $INT_NET -j DROP

### ACCEPT rules
$IPTABLES -A FORWARD -p tcp -i eth1 -s $INT_NET --dport 21 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp -i eth1 -s $INT_NET --dport 22 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp -i eth1 -s $INT_NET --dport 25 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp -i eth1 -s $INT_NET --dport 43 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp --dport 80 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp --dport 443 --syn -m state --state NEW -j ACCEPT
```

```
$IPTABLES -A FORWARD -p tcp -i eth1 -s $INT_NET --dport 4321 --syn -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p tcp --dport 53 -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p udp --dport 53 -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -p icmp --icmp-type echo-request -j ACCEPT

### default LOG rule
$IPTABLES -A FORWARD -i ! lo -j LOG --log-prefix "DROP " --log-ip-options --log-tcp-options

###### NAT rules ######
#
echo "[+] Setting up NAT rules..."
$IPTABLES -t nat -A PREROUTING -p tcp --dport 80 -i eth0 -j DNAT --to 192.168.10.3:80
$IPTABLES -t nat -A PREROUTING -p tcp --dport 443 -i eth0 -j DNAT --to 192.168.10.3:443
$IPTABLES -t nat -A PREROUTING -p udp --dport 53 -i eth0 -j DNAT --to 192.168.10.4:53
$IPTABLES -t nat -A POSTROUTING -s $INT_NET -o eth0 -j MASQUERADE

###### forwarding ######
#
echo "[+] Enabling IP forwarding..."
echo 1 > /proc/sys/net/ipv4/ip_forward

exit
### EOF ###
```