AUUG Systems Administration Symposium

9 April 2003

Abusing the Debian **ipmasq** package

Martin Schwenke

IBM OzLabs Linux Technology Center

<martins@au.ibm.com> · <martin@meltin.net>

The plan. . .

- The Debian ipmasq package.

The plan. . .

- The Debian ipmasq package.
- Abusing it:
  - Connection tracking firewall.

The plan. . .

- The Debian ipmasq package.
- Abusing it:
  - Connection tracking firewall.
  - Automatic SOCKSification.

The plan. . .

- The Debian ipmasq package.
- Abusing it:
  - Connection tracking firewall.
  - Automatic SOCKSification.
  - IP accounting.

The plan. . .

- The Debian ipmasq package.
- Abusing it:
  - Connection tracking firewall.
  - Automatic SOCKSification.
  - IP accounting.
  - Destination NAT.

The Debian **ipmasq** package

- In short:

The Debian **ipmasq** package

- In short:
  1. `apt-get install ipmasq`

The Debian **ipmasq** package

- In short:
  1. `apt-get install ipmasq`
  2. `ipmasq`

The Debian **ipmasq** package

- In short:
  1. `apt-get install ipmasq`
  2. `ipmasq`
- This turns the box into an masquerading router:

The Debian **ipmasq** package

- In short:
  1. `apt-get install ipmasq`
  2. `ipmasq`
- This turns the box into an masquerading router:
  - Internal networks can talk to each other.
  - Connections from internal networks to the external network are masqueraded (source NATed).
  - Connections to the external interface are allowed.

The Debian **ipmasq** package

- In short:
  1. `apt-get install ipmasq`
  2. `ipmasq`
- This turns the box into an masquerading router:
  - Internal networks can talk to each other.
  - Connections from internal networks to the external network are masqueraded (source NATed).
  - Connections to the external interface are allowed.
- No configuration required!

How does **ipmasq** work?

- You see, there are these 'rules' files:

```
# cd /etc/ipmasq/rules
# ls [AFIMOZ]*.def
A00path.def                I10lo.def         O30intbcast.def
A00sanitycheck.def         I15lospoof.def    O30internal.def
A01interfaces.def          I30intbcast.def   O32intmcast.def
A01mungeexternal.def       I30internal.def   O70masq.def
A01precompute.def          I32intmcast.def   O90extbcast.def
A02masqmethod.def          I70masq.def       O90external.def
A02unkernelforward.def     I90extbcast.def   Z90kernelforward.def
A03flush.def               I90external.def   Z92timeouts.def
A04functions.def           M70masq.def       Z99ipmasqrules.def
F30internal.def            O10lo.def         ZZZdenyandlog.def
```

What do the ‘rules’ files do?

A*: setup — external/internal interfaces, method, forwarding off.
F*: forwarding — between internal interfaces.
I*: input — loopback, anti-spoofing, internal/external interfaces.
M*: masquerading — internal to external.
O*: output — loopback, internal/external interfaces.
Z*: finale — forwarding on, deny/log.

What do the ‘rules’ files do?

`A*`: setup — external/internal interfaces, method, forwarding off.

`F*`: forwarding — between internal interfaces.

`I*`: input — loopback, anti-spoofing, internal/external interfaces.

`M*`: masquerading — internal to external.

`O*`: output — loopback, internal/external interfaces.

`Z*`: finale — forwarding on, deny/log.

Examples shown as we change things. . .

head -5 A00path.def

```
# You should not edit this file.  Instead, create a file with the same
# name as this one, but with a .rul extension instead of .def.  The
# .rul file will override this one.
#
# However, any changes you make to this file will be preserved.
```

Connection tracking firewall — steps

`A10netfilteronly.rul`: Ensure we're running **netfilter**.

`I90extbcast.rul`: No incoming external broadcasts.

`I90external.rul`: Only allow desired incoming external packets.

`M70masq.rul`: Connection track forwarded connections.

`ZZZdenyandlog.rul`: Don't log boring things.

cat A10netfilteronly.rul

```
#: Only support MASQMETHOD = netfilter.
if [ "$MASQMETHOD" != "netfilter" ] ; then
        echo "MASQMETHOD \"$MASQMETHOD\" not supported" 1>&2
        exit 1
fi
```

tail +7 I90extbcast.def

```
#: Accept dumb broadcast packets on external interfaces
if [ -n "$EXTERNAL_IN" ]; then
    for i in $EXTERNAL_IN; do
        ipnm_cache $i
        case $MASQMETHOD in
        ipfwadm)
            $IPFWADM -I -a accept -W ${i%%:*} -D 255.255.255.255/32
            ;;
        ipchains)
            $IPCHAINS -A input -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
            ;;
        netfilter)
            $IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
            ;;
        esac
    done
fi
```

cat I90extbcast.rul

```
#: *DON'T* accept dumb broadcast packets on external interfaces
if [ -n "$EXTERNAL_IN" ]; then
    for i in $EXTERNAL_IN; do
        ipnm_cache $i
        case $MASQMETHOD in
        ipfwadm)
            $IPFWADM -I -a accept -W ${i%%:*} -D 255.255.255.255/32
            ;;
        ipchains)
            $IPCHAINS -A input -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
            ;;
        netfilter)
            : #$IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
            ;;
        esac
    done
fi
```

diff -u I90extbcast.{def,rul}

```
--- I90extbcast.def     2003-04-01 11:42:28.000000000 +1000
+++ I90extbcast.rul     2003-04-01 11:43:59.000000000 +1000
@@ -1,10 +1,4 @@
-# You should not edit this file.  Instead, create a file with the same
...
-#: Accept dumb broadcast packets on external interfaces
+#: DON'T accept dumb broadcast packets on external interfaces
 if [ -n "$EXTERNAL_IN" ]; then
     for i in $EXTERNAL_IN; do
        ipnm_cache $i
@@ -16,7 +10,7 @@
            $IPCHAINS -A input -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
             ;;
        netfilter)
-           $IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
+           : #$IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d 255.255.255.255/32
            ;;
         esac
     done
```

diff -u I90external.{def,rul}

```
--- I90external.def     2003-04-01 12:13:01.000000000 +1000
+++ I90external.rul     2002-07-25 16:24:19.000000000 +1000
@@ -22,10 +16,26 @@
            fi
             ;;
        netfilter)
-           $IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d $IPOFIF/32
-           if [ -n "$BCOFIF" ]; then
-               $IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d $BCOFIF/32
-           fi
+           # Allow continuations of existing connections in.
+           $IPTABLES -A INPUT -j ACCEPT \
+                -i ${i%%:*} -m state --state ESTABLISHED,RELATED
+
+           # Allow new connections to some TCP ports.
+           for p in ssh smtp http rsync ; do
+               $IPTABLES -A INPUT -j ACCEPT \
+                   -i ${i%%:*} -p tcp -d ${IPOFIF} --dport ${p} \
+                   -m state --state NEW
+           done
...
```

diff -u I90external.{def,rul} #...

```
--- I90external.def     2003-04-01 12:13:01.000000000 +1000
+++ I90external.rul     2002-07-25 16:24:19.000000000 +1000
@@ -22,10 +16,26 @@
...
+           # Make the ident lookup on some mail/web servers fail quickly.
+           $IPTABLES -A INPUT -j REJECT \
+               -i ${i%%:*} -p tcp -d ${IPOFIF} --dport ident \
+               -m state --state NEW
+
+           # Allow pings.  All other ICMP stuff should work due to conntrack.
+           $IPTABLES -A INPUT -j ACCEPT -i ${i%%:*} -d $IPOFIF \
+               -p icmp --icmp-type ping \
+               -m state --state NEW
            ;;
         esac
     done
```

diff -u M70masq.{def,rul}

```
--- M70masq.def 2003-04-01 12:24:29.000000000 +1000
+++ M70masq.rul 2003-02-03 12:04:46.000000000 +1100
...
@@ -22,13 +16,15 @@
                ;;
            netfilter)
                if [ -n "$PEEROFIF" ]; then
...
                else
                    $IPTABLES -t nat -A POSTROUTING -j MASQUERADE \
-                       -s $IPOFIF/$NMOFIF
+                       -o ${j%%:*} -s $IPOFIF/$NMOFIF
                    $IPTABLES -A FORWARD -j ACCEPT \
                        -i ${i%%:*} -o ${j%%:*} -s $IPOFIF/$NMOFIF
                    $IPTABLES -A FORWARD -j ACCEPT \
-                       -o ${i%%:*} -i ${j%%:*} -d $IPOFIF/$NMOFIF
+                       -o ${i%%:*} -i ${j%%:*} -d $IPOFIF/$NMOFIF \
+                       -m state --state ESTABLISHED,RELATED
                fi
                ;;
            esac
```

diff -u ZZZdenyandlog.{def,rul}

```
--- ZZZdenyandlog.def   2002-01-02 08:26:05.000000000 +1100
+++ ZZZdenyandlog.rul   2002-07-25 16:32:27.000000000 +1000
@@ -17,6 +17,23 @@
     $IPCHAINS --no-warnings -A forward -j DENY -s 0.0.0.0/0 -d 0.0.0.0/0 -l
     ;;
 netfilter)
+    # DROP some ports for UDP, but do it silently.
+    for p in bootps netbios-ns netbios-dgm snmp who route 1985; do
+       $IPTABLES -A INPUT -j DROP -p udp --destination-port ${p}
+    done
...
+
+    $IPTABLES -A INPUT -j DROP -d 255.255.255.255/32
+
+    # Weird crap from XXX.austin.ibm.com is filling our logs!
+    $IPTABLES -A INPUT -j DROP -p tcp \
+       -s 9.3.165.66 -d 9.185.116.201 --dport 1021:1022
+
     $IPTABLES -A INPUT -j LOG -s 0.0.0.0/0 -d 0.0.0.0/0
     $IPTABLES -A INPUT -j DROP -s 0.0.0.0/0 -d 0.0.0.0/0
     $IPTABLES -A OUTPUT -j LOG -s 0.0.0.0/0 -d 0.0.0.0/0
```

Automatic SOCKSification — steps

`A05autosocksify.def`: variables — `autosocksifyd` port, SOCKSified ports file, local address file.

`I20autosocksify.def`: redirect certain incoming internal connections to `autosocksifyd`.

`O80autosocksify.def`: redirect certain outgoing connections to `autosocksifyd`.

autosocksifyd?

-
  ```
  $ grep autosocksify /etc/inetd.conf
  11080 stream tcp nowait.256 root /usr/sbin/tcpd /usr/sbin/autosocksifyd
  ```

autosocksifyd?

-
  ```
  $ grep autosocksify /etc/inetd.conf
  11080 stream tcp nowait.256 root /usr/sbin/tcpd /usr/sbin/autosocksifyd
  ```

-
  ```
  $ cat /usr/sbin/autosocksifyd
  #!/bin/sh

  set -- $(/usr/sbin/nf_getsockname -n)
  dstaddr=$1
  dstport=$2

  exec /usr/bin/socksify \
      /usr/bin/redir --inetd --caddr=${dstaddr} --cport=${dstport}
  ```

nf_getsockname()?

- nf_getsockname = tcputils-0.6.2/getpeername.c + nf_getsockname()

nf_getsockname()?

- nf_getsockname = tcputils-0.6.2/getpeername.c + nf_getsockname()

-
  ```
  /* nf_getsockname() - netfilter SO_ORIGINAL_DST variant of getsockopt()
   *
   * Within the new Linux netfilter framework, NAT functionality is cleanly
   * separated from the TCP/IP core processing. In old days, you could easily
   * retrieve the original destination (IP address and port) of a transparently
   * proxied connection by calling the normal getsockname() syscall.
   * With netfilter, getsockname() returns the real local IP address and port.
   * However, the netfilter code gives all TCP sockets a new socket option,
   * SO_ORIGINAL_DST, for retrieval of the original IP/port combination.
   *
   * This file implements a function nf_getsockname(), with the same calling
   * convention as getsockname() itself; it uses SO_ORIGINAL_DST, and if that
   * fails, falls back to using getsockname() itself.
   *
   * Public domain by Patrick Schaaf <bof@bof.de>
   */
  ```

cat A05autosocksify.def

```
# You should not edit this file.  Instead, create a file with the same
# name as this one, but with a .rul extension instead of .def.  The
# .rul file will override this one.
#
# However, any changes you make to this file will be preserved.

autosocksify_inetd=/usr/sbin/autosocksifyd
autosocksify_port=11080
autosocksify_ports=/etc/autosocksify/ports
autosocksify_local=/etc/autosocksify/local
```

cat /etc/autosocksify/ports

```
# List of destination TCP ports that you want to autoSOCKSify.  You
# can list any ports that iptables will understand.  That is, things
# in /etc/services are allowed.
#
# Sensible comment syntax is allowed: s/#.*$//
ssh
smtp
whois
http https
nntp
rsync
cvspserver
dict
5000 5005 14690         # BitKeeper
ircd ircs
11371                   # PGP/GPG key server, doesn't seem work :-(
```

cat /etc/autosocksify/local

```
# List of destination network addresses that you don't want to
# autoSOCKSify, such as 192.168.1.1/24, separated by whitespace.
#
# Sensible comment syntax is allowed: s/#.*$//
10.61.2.0/24            # OzLabs
9.0.0.0/8               # IBM
146.84.0.0/16           # Tivoli
138.95.0.0/16           # Sequent
```

cat I20autosocksify.def

```
if [ -x "$autosocksify_inetd" -a -n "$INTERNAL" ]; then
    for i in $INTERNAL; do
        ipnm_cache $i
        case $MASQMETHOD in
        ipfwadm|ipchains)
            echo "error!" 1>&2 ; exit 1 ;;
        netfilter)
            for p in $([ -r $autosocksify_ports ] && \
                       sed -e 's/#.*$//' $autosocksify_ports) ; do
                for a in $([ -r $autosocksify_local ] && \
                           sed -e 's/#.*$//' $autosocksify_local) ; do
                    $IPTABLES -t nat -A PREROUTING -j ACCEPT \
                        -i ${i%%:*} -p tcp -d $a --dport $p
                done
                $IPTABLES -t nat -A PREROUTING -j REDIRECT \
                    -i ${i%%:*} -p tcp --dport $p --to-ports $autosocksify_port
            done
            ;;
        esac
    done
fi
```

cat 080autosocksify.def

```
if [ -x "$autosocksify_inetd" -a -n "$EXTERNAL_OUT" ]; then
    for i in $EXTERNAL_OUT; do
        ipnm_cache $i
        case $MASQMETHOD in
        ipfwadm|ipchains)
            echo "error!" 1>&2 ; exit 1 ;;
        netfilter)
            for p in $([ -r $autosocksify_ports ] && \
                       sed -e 's/#.*$//' $autosocksify_ports) ; do
                for a in $([ -r $autosocksify_local ] && \
                           sed -e 's/#.*$//' $autosocksify_local) ; do
                    $IPTABLES -t nat -A OUTPUT -j ACCEPT \
                        -o ${i%%:*} -p tcp -d $a --dport $p
                done
                $IPTABLES -t nat -A OUTPUT -j REDIRECT \
                    -o ${i%%:*} -p tcp --dport $p --to-ports $autosocksify_port
            done
            ;;
        esac
    done
fi
```

IP accounting — steps

- All in `C00ipacct.def`.
- Accounting chains: `acctin` & `acctout`.

1. Dump accounting chains.
2. (Re)create accounting chains.
3. Add accounting rules to accounting chains.
4. Add accounting chains to `INPUT`/`OUTPUT`, `FORWARD`/`FORWARD` chains for incoming/outgoing traffic on external interfaces.

tail +10 C00ipacct.def | head -22

```
netfilter)
    [ -x /usr/sbin/ipacct-dump ] && /usr/sbin/ipacct-dump

    $IPTABLES -N acctin >/dev/null 2>&1
    $IPTABLES -N acctout >/dev/null 2>&1
    $IPTABLES -F acctin >/dev/null 2>&1
    $IPTABLES -F acctout >/dev/null 2>&1

    # Default rules that count everything.  Insert more specific rules
    # *after* these.
    $IPTABLES -A acctin
    $IPTABLES -A acctout

    if [ -n "$EXTERNAL" ]; then
        for i in $EXTERNAL; do
            $IPTABLES -A INPUT   -i ${i%%:*} -j acctin
            $IPTABLES -A OUTPUT  -o ${i%%:*} -j acctout
            $IPTABLES -A FORWARD -i ${i%%:*} -j acctin
            $IPTABLES -A FORWARD -o ${i%%:*} -j acctout
        done
    fi
```

diff -u C00ipacct.def /tmp/C00ipacct.1

```
--- C00ipacct.def       2003-04-01 16:25:10.000000000 +1000
+++ /tmp/C00ipacct.1    2003-04-01 16:26:55.000000000 +1000
@@ -15,6 +15,15 @@
     $IPTABLES -F acctin >/dev/null 2>&1
     $IPTABLES -F acctout >/dev/null 2>&1

+    ############################################################
+
+    # Stuff that we don't want to account.  This outbound traffic only
+    # exists internally and is regenerated by the redir job.
+    $IPTABLES -A acctout -j RETURN \
+       -s 127.0.0.1/32 -d 127.0.0.1/32 -p tcp --dport 11080
+
+    ############################################################
+
     # Default rules that count everything.  Insert more specific rules
     # *after* these.
```

diff -u C00ipacct.def /tmp/C00ipacct.2

```
--- C00ipacct.def       2003-04-01 16:25:10.000000000 +1000
+++ /tmp/C00ipacct.2    2003-04-01 16:29:40.000000000 +1000
@@ -21,6 +21,18 @@
     $IPTABLES -A acctin
     $IPTABLES -A acctout

+    # Local accounting rules.
+    for ip in \
+       9.190.161.0/24 9.190.162.0/24 9.190.163.0/24 9.190.164.0/24 \
+       9.190.250.32/32 9.190.250.93/32 9.190.250.94/32 9.139.253.253 \
+       9.190.0.0/16 \
+       9.0.0.0/8 \
+       ; do
+
+       $IPTABLES -A acctin  -s $ip -j RETURN
+       $IPTABLES -A acctout -d $ip -j RETURN
+    done
+
     if [ -n "$EXTERNAL" ]; then
        for i in $EXTERNAL; do
           $IPTABLES -A INPUT   -i ${i%%:*} -j acctin
```

Destination NAT — steps

- All in `M85dnatter.rul`.

1. Destination NAT certain external connections to an internal address, based on source address and (maybe) destination port.
2. Tell the `FORWARD` chain that this is OK.

cat /etc/dnatter/dnatter.conf

```
# Each line describes a DNAT of one port from this router to a target host.
# Separator is whitespace.
# 1st field is source address/network.
# 2nd field is DNAT target host.
# 3rd field is destination port on this router.  This can be "ALL".
# 4th field (optional, defaults to value of 3rd field) is DNAT target port.

9.3.197.245/32 10.61.2.86      999 22  # austin  -> avago [martins/test]
9.47.18.0/24   10.61.2.86      22      # rasdiag -> avago [martins]
9.186.129.8/32 10.61.2.33      22      # Suparna -> superego
```

tail +25 M85dnatter.rul | head -21

```
sed -e 's/[ \t]*#.*$//' -e '/^[ \t]*$/d' $dnatter_conf |
while read src dst lport dport ; do
    if [ -z "$lport" -o "$lport" = "ALL" ] ; then
        $IPTABLES -t nat -A PREROUTING \
            -i ${i%%:*} -s $src -p tcp \
            -j DNAT --to-destination $dst
        $IPTABLES -A FORWARD -j ACCEPT \
            -i ${i%%:*} -s $src -p tcp \
            -d $dst \
            -m state --state NEW
    else
        [ "$dport" ] || dport=$lport
        $IPTABLES -t nat -A PREROUTING \
            -i ${i%%:*} -s $src -p tcp --dport $lport \
            -j DNAT --to-destination "${dst}:${dport}"
        $IPTABLES -A FORWARD -j ACCEPT \
            -i ${i%%:*} -s $src -p tcp \
            -d $dst --dport $dport \
            -m state --state NEW
    fi
done
```

Questions?

?