Building a Linux Firewall with IPTABLES

Think-Linux Conference

Scott D. Courtney
Senior Engineer, Sine Nomine Associates
43596 Blacksmith Square; Ashburn VA 20147-4606
October, 2002 http://sinenomine.net/

IP Networking Concepts: Addresses

- In *conceptual* terms, an IP address represents a given device ("host") on the network.
- IP addresses are written as dotted-decimal numbers, such as 192.168.64.4 or 10.0.3.2.
- The IP address gets packets (messages) properly transferred from one host to another, but does not imply anything about their content or meaning or which process should receive them.

IP Networking Concepts: Ports

- Once a packet reaches the destination host, it is sent to a specified "port" on that host.
- There are about 65,000 available numbers per host. The first 1024 are reserved to privileged processes such as daemons.
- "Well-known" port numbers are used for ubiquitous services such as Telnet, FTP, HTTP, and so on, at the server end.
- The file /etc/services defines well-known port numbers.
- Randomly-assigned port numbers (above the first 1024) are used at the client end of most IP applications.
- Source & destination IP, plus source & destination port, together identify a unique socket.

IP Networking Concepts: Protocols

- The protocol represents what “kind” of traffic is being sent across the network.
- TCP (Transmission Control Protocol) maintains a connection (channel) between two hosts.
- UDP (User Datagram Protocol) sends data statelessly, without establishing a connection.
- ICMP (Internet Control Message Protocol) handles administrative functions such as PING.
- There are others, outside the scope of this class.

IP Networking Concepts: General

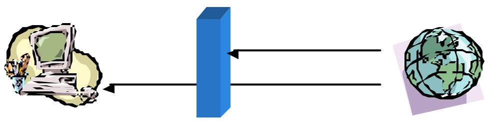
- It takes a source address, a destination address, a source port, a destination port, and a protocol type to characterize traffic for firewalling purposes.
- Additional parameters, such as new vs. existing connection (for TCP traffic) or QoS requests, can be included in more sophisticated firewalls.
- Notation is typically address:port, e.g. 192.168.64.4:80 with protocol specified separately
- Masks specify ranges, e.g. 192.168.64.0/24 means the same as 192.168.64.*

What is Packet Filtering, and Why Do I Need It?

- Blocking unwanted traffic or probes from outside
- Limiting access to Internet from certain hosts
- Network Address Translation (NAT): sharing a single Internet address with multiple hosts from an internal LAN
- Redirecting specific inbound requests to selected internal hosts
- Rewriting attributes of packets for Quality of Service (QoS) or other sophisticated filtering
- **Packet filtering will not stop viruses, etc.!**

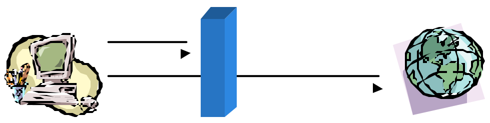
What Is Packet Filtering, and Why Do I Need It?

- Blocking unwanted traffic or probes from outside
  - By IP address (you may wish to trust specific hosts)
  - By destination port (allowing specific services, such as HTTP, but excluding all others)
  - By protocol type (e.g., disallowing all PINGs from outside)



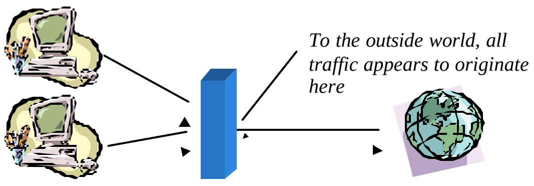
What Is Packet Filtering, and Why Do I Need It?

- Limiting access to Internet from certain hosts
  - By IP address (allow kiosks to access only *intranet* from web browser, for example)
  - By destination port (allowing specific services, such as HTTP, but excluding all others)
  - Typically political/management, not "security" *per se*



What Is Packet Filtering, and Why Do I Need It?

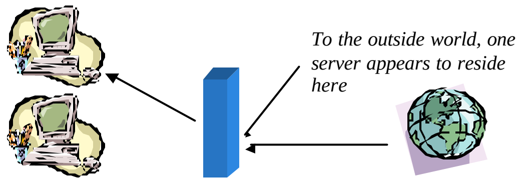
- Network Address Translation (NAT)
  - Many ISPs grant only a single (often dynamic) IP
  - In a *limited* way, NAT helps to make LAN hosts harder to access from outside


To the outside world, all traffic appears to originate here

What Is Packet Filtering, and Why Do I Need It?

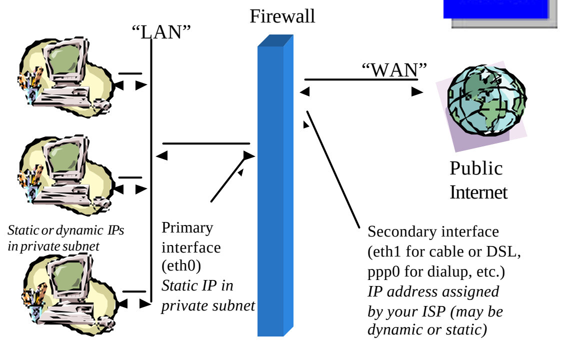
Sine Nomine Associates

- Inbound port redirection
  - Allows *specific* connection to your LAN from outside
  - Use with **extreme** caution: If the one host is compromised, intruders may access your entire LAN



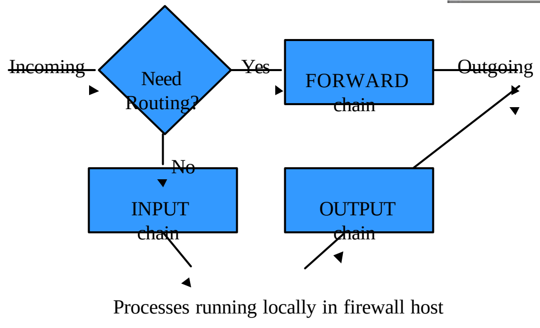
Anatomy of a Linux Firewall

Sine Nomine Associates


"LAN"
Firewall
"WAN"
Public Internet
Static or dynamic IPs in private subnet
Primary interface (eth0) *Static IP in private subnet*
Secondary interface (eth1 for cable or DSL, ppp0 for dialup, etc.) *IP address assigned by your ISP (may be dynamic or static)*

Packet Flow Through Filters (Table "filter")

Sine Nomine Associates



Software Components of Linux Packet Filtering

- Kernel support: in “Network Options” enable “Packet Filtering”
- Kernel loadable modules (in “Network Options / IP: Netfilter Configuration”)
  - Connection tracking (for NAT/masquerade)
  - FTP and IRC protocols (special cases)
  - IPTABLES support
    - Packet filtering, REJECT, Full NAT, MASQUERADE, REDIRECT
    - Others as your needs dictate
- IPTABLES command
  - Controls the kernel-space filtering, but does do the work itself
  - Not a daemon, and runs in user-space

Basic IPTABLES Operations

• -I *number*	Insert a new rule before rule *number*
• -A	Append a new rule at end of chain
• -R *number*	Replace rule *number* with new rule
• -D *number*	Delete rule *number*
• -F	Flush the chain (delete all rules)
• -N *chain*	New chain (specify name)
• -X *chain*	Delete user-defined *chain*
• -L *chain*	List the rules in *chain*

Note: Rule “1” is the first rule in each chain.

IPTABLES Parameters

- Matching parameters

-p *protocol*	Matches specified protocol
-s *source*	Matches source address
	Can use mask, as in “192.168.0.0/16”
	Can precede *source* with “!” to invert match
-d *destination*	Matches destination address (like -s)
-i *input_intfc*	Packets arriving on this interface
-o *output_intfc*	Packets departing on this interface
-f	Matches fragments (“! -f” matches head, or unfragmented, packets)

IPTABLES Parameters

- Targets
  -j *target* Jump to *target* (chain or predefined)
- Targets include (among others)

– LOG	Make a log entry (otherwise no-op)
– REJECT	Send back an error response
– DROP	Ignore packet without responding
– SNAT	Source network address translation
– DNAT	Destination network address translation
– MASQUERADE	Source NAT in a dialup context
– REDIRECT	Destination set to local (firewall) host

IPTABLES Parameters

- Specifying the table
  -t *table*
- Tables include

– “filter”	(the default)
– “nat”	(NAT and masquerade)
– “mangle”	(rewrite packets)

- “mangle” table is not covered in this presentation

IPTABLES Parameters

- Stateful filtering parameters
  -m state Causes matching on state of traffic
  --state

NEW	New communication request
ESTABLISHED	Reply to previous packet
RELATED	Like ESTABLISHED, but for special cases where the packet is not strictly a reply packet

IPTABLES Parameters

- More matching options for " tcp" and "udp" protocols only:
  --source-port *[!] port[:port]*
  Examples: --source-port 0:1023
  --source-port ! 80
  --destination-port *[!] port[:port]*
- --sport and --dport are synonyms for the above
- Matching options for "tcp" protocol only:
  --syn
  Matches only a packet that is requesting a new TCP connection

Building a Basic Firewall (Blocks all inbound WAN connects)

- Assume eth0 is LAN, eth1 is WAN (DSL or cable) for these examples (use ppp0 instead of eth1 for dialup)
- LAN addresses are 192.168.*x.y* and firewall is 192.168.1.1 for these examples
- Build a new chain to filter on packet state:

```
iptables -N block
iptables -A block -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A block -m state --state NEW -i ! eth1 -j ACCEPT
iptables -A block -j DROP
```

- Link to that new chain:

```
iptables -A INPUT -j block
iptables -A FORWARD -j block
```

Building a Basic Firewall (Selective By Ports)

- These are *example* rules, independent of previous slide
- Block connections from eth1 on privileged ports (inserts new rules 1 and 2 on input chain)

```
iptables -I INPUT 1 --dport 0:1023 -i eth1 -p tcp -j DROP
iptables -I INPUT 2 --dport 0:1023 -i eth1 -p udp -j DROP
```

- Allow inbound web connection even from Internet (note that this rule inserts *ahead* of the other two)

```
iptables -I INPUT 1 --dport 80 -p tcp -i eth1 -j ACCEPT
```

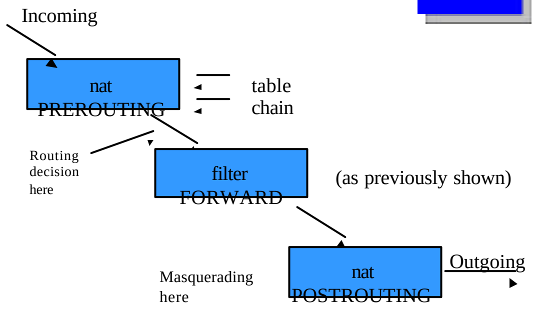
- Allow inbound SSH connection from one trusted host (note that this rule inserts *ahead* of the others)

```
iptables -I INPUT 1 --dport 22 -p tcp -s 123.45.67.89 -j ACCEPT
```

Turning on IPv4 Routing

- Need to do this if masquerading
- As root,
  `echo 1 > /proc/sys/net/ipv4/ip_forward`
- Don’t do this until after your firewall rules are added
- Use private IP subnets (192.168.*x*.*y*, 172.16.*x*.*y*, or 10.*x*.*y*.*z*) on your LAN for additional safety (most ISPs will not route these subnets).
- Don’t rely on the private subnet for “real” security!
- Turn off with
  `echo 0 > /proc/sys/net/ipv4/ip_forward`

Masquerading Packet Flow



Adding Masquerade

- Add masquerading for all outbound connections (dynamic WAN address)

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.0.0/16 \
     -j MASQUERADE
```

- Add masquerading for all outbound connections (static WAN address 12.34.56.78)

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.0.0/16 \
     -j SNAT --to 12.34.56.78
```

- MASQUERADE “forgets” connections when WAN goes down, while SNAT does not.
- Common ***mistake*** (serious security hole)

```
iptables -t nat -A POSTROUTING -j MASQUERADE
```

  This works from your LAN, but allows outsiders to masquerade through *your* firewall.

Inbound Redirection

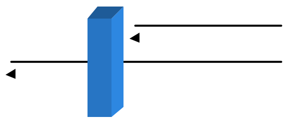
- Assume WAN IP is 12.34.56.78
- Allow inbound web access to be redirected to host 192.168.10.54

```
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 80 -m state \
     --state NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A PREROUTING -t nat -p tcp -d 12.34.56.78 \
     --dport 80 -j DNAT --to 192.168.10.54:80
```

- It is rare that you would do this with a dynamic WAN address
- Normally you would have an externally visible server on a DMZ segment, which might be (for example) eth2
- Use this technique with *extreme* caution, especially if the web server has access onto your LAN!

Testing Your Firewall From Outside

- Have a friend “tiger team” probe you, using NMAP or a similar utility, from outside your own LAN.
- For a rudimentary test, use Steve Gibson’s “Shields Up” probe (http://www.grc.com/)
- Very important step! It is easy to make a small error that has big consequences.



Conclusion

- Linux and IPTABLES can make a remarkably capable firewall at little or no cost
- There is much more to learn about IPTABLES than the things covered in this session
- Type “man iptables” to get syntax help
- Read the HOWTO and other documentation
- Example rc.firewall scripts (SysV Init) are on the web
- Have fun, and experiment!

Kernel Loadable Modules Reference (Partial List)

ip_conntrack.o	Connection tracking
ip_conntrack_ftp.o	FTP connection tracking
ip_conntrack_irc.o	IRC connection tracking
ip_tables.o	IPTABLES support
ipt_MASQUERADE.o	MASQUERADE target
ipt_REDIRECT.o	REDIRECT target
ipt_nat.o	NAT support
iptable_filter.o	General filtering support
ipt_nat_ftp.o	NAT of FTP protocol
ipt_nat_irc.o	NAT of IRC protocol

Reference: Private IP Addresses

- The Internet Engineering Task Force (IETF) defines three IP address ranges for private networks.
  - 10.0.0.0 with 8-bit netmask (one Class A)
  - 172.16.0.0 with 12-bit netmask (multiple Class B)
  - 192.168.0.0 with 16-bit netmask (multiple Class C)
- The 172.16 range contains 16 Class B networks
- The 192.168 range contains 256 Class C networks
- RFC1918 states that these are not to be routed on public network, but some ISPs may not fully comply
- The full text of RFC1918 is at http://www.ietf.org/rfc/rfc1918.txt

Webliography

- **Netfilter / IPTABLES Home Page**
  http://www.netfilter.org/ or http://www.iptables.org/
- **IPTABLES Tutorial**
  http://www.linuxvoodoo.com/howto/iptables/iptables-tutorial.html
- **Packet Filtering HOWTO**
  http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html
- **IP Masquerading HOWTO**
  http://www.e-infomax.com/ipmasq/howto/c-html/
- **IP Masquerading Resource Site**
  http://www.e-infomax.com/ipmasq/
- **Specialized Add-On Modules**
  http://www.e-infomax.com/ipmasq/matrix24.html