Firewalling

Alessandro Barenghi

Dipartimento di Elettronica e Informazione
Politecnico di Milano

barenghi - at - elet.polimi.it

May 19, 2011

Recap

By now , you should be familiar with...

- Programming with sockets employing different protocols
- System programming, synchronization primitives and IPC
- System administration skills , as far as the local host and network monitoring go

Lesson contents

Overview

- Netfilter/Iptables Structure
- Policy construction
- Rules setting
- Advanced matching

Packet filtering

What's in a Firewall...

- A firewall (or packet filter) is a toolkit deciding whether packets passing from an host are to be kept or discarded
- Structurally :
  - Integrated with the network stack as much as possible
  - Usually the packet filtering is in kernelspace, mainly due to performance reasons
  - Firewall management tools usually reside in userspace, due to ease of use
- We will examine the NetFilter (kernelspace) / Iptables (userspace) packet filtering suite

Packet filtering

Where?

The (main) firewall should be the single point of contact between the secure and insecure zone

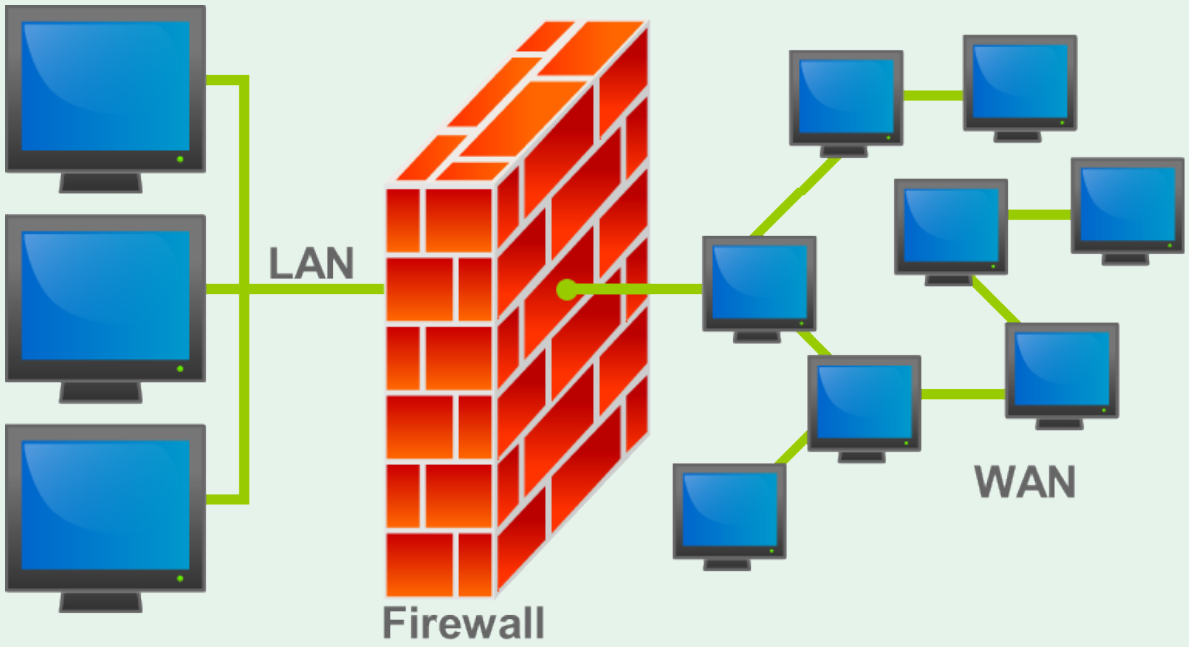

Figure: Firewall Placement

Packet filtering

Why firewalling?

- Avoiding unauthorized connections regardless of the availability of a server
- Packet sanitization (checksum check) can be performed during filtering
- Stateful packet filtering also enforces observance of Level 3+ protocols
- Network and Port Address translation strategies can be employed by a packet-mangling firewall

Netfilter Structure

Overview

- NetFilter is a set of modules implementing filtering functions
- The NetFilter structure is based on five hooks, placed on the path of incoming/outgoing packets
- The communication with the userspace management tools happens via Netlink sockets
- Each of the five hooks executes a set of rule each time a packet passes through it

Structure

Overview

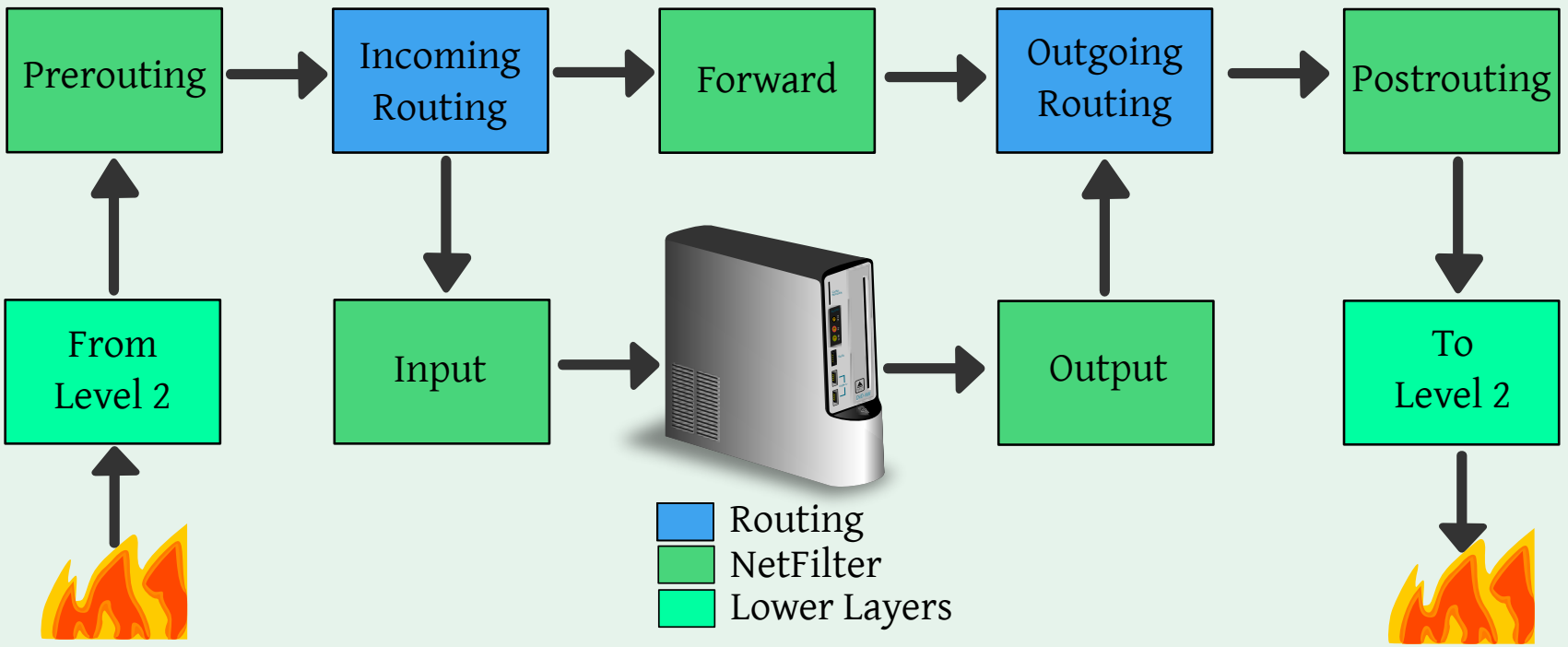

Figure: NetFilter Structure

Netfilter chains

Overview

- A Netfilter chain is characterised by an ordered list of rules which are triggered on a certain condition on the packet
- If no rule matches the packet, the default action, i.e. the chain policy is adopted
- Up to four tables containing chains are present (filter,nat, mangle and raw) for each Netfilter hook
- It is possible to create custom chains of rules in order to avoid the crowding of the default chain
- There is no possibility to add hook structures by default (obviously, you can write an extra module :))

Hook policies

Setting the defaults

- Every builtin chain has a default policy, i.e. a default action to be performed on the packet
  - `ACCEPT`: the packet flows through the hook, towards its destination
  - `QUEUE`: the packet is sent to the userspace via Netlink for examination
  - `DROP`: the packet is discarded and treated as it never existed
- A hook policy can be set up with `iptables -P <chain> <policy>`
- The default policy, with which the kernel boots Netfilter is `ACCEPT` for all the base chains

Hook policies

Reasonable policies

- Reasonable policies usually are :
  - `PRE/POSTROUTING`: set to `ACCEPT`, these chains are not meant for dropping
  - `INPUT`: set to `DROP`, whitelist is better than blacklist
  - `FORWARD`: set to `DROP`, "Thou shall not pass" is a reasonable default for the same reasons
  - `OUTPUT`: set to `ACCEPT`, although particularly restrictive policies may need a `DROP`

Rules - management

Rule structure

- The Netfilter behaviour is modified via the `iptables` command
- A rule is composed of two parts, the match and the target
- The match specifies the conditions regarding the packet which will trigger the rule
- The target specifies the fate of the packet
- For basically all match specifications , prepending a `!` mark inverts the match

Rules - management

Targets

- Possible targets (with extensions) for a rule are :
  - `ACCEPT/DROP` : behave exactly as the policies
  - `REJECT`: The packet is dropped but, if allowed by the protocol, the sender is notified of the rejection
  - `LOG`: A line in the kernel log is written, and the check on the chain of rules goes on
  - `MIRROR`: Swaps source and destination address and immediately sends the packets without passing via the other chains
  - `RATEEST`: adds this packet to the statistic of a rate estimator, then the chain checks go on

Rules - management

Rule management

- The generic `iptables` command is structured as : `iptables [-t table] <action> <rule>`
- Possible actions are :
  - `-A <chain>` : appends a rule at the end of the chain
  - `-D <chain>` : deletes the specific rule (the number of the rule may be indicated instead)
  - `-I <chain> <num>`: inserts the rule as the n-th
  - `-R <chain> <num>`: replaces the n-th rule
  - `-L`: lists all the rules of a chain
  - `-F`: flushes a chain (but does not reset the policy to `ACCEPT`)

Rules - 1

Matching interfaces

- The first and most simple match for a packet is to decide an action depending on the interface it was received on
- The inbound/outbound interface matches are specified via the `-i <iface>/-o <iface>` option
- The `-i/-o` options are limited to some chains, namely:
  - `-i` can only be used in `INPUT`, `FORWARD` and `PREROUTING`
  - `-i` can only be used in `OUTPUT`, `FORWARD` and `POSTROUTING`
- The most common use of this match is to differentiate the reasonably trusted zone of the network (LAN side) from the really untrusted side (WAN side)

Rules - 1

Matching interfaces - 2

- A special case for interface matching is the loopback interface `lo`
- This interface should never be filtered, lest a couple of applications *will* misbehave
- Accepting all packets with destination address equal to 127.0.0.1 is not equivalent to accepting `lo` (See RFC3330)
- Accepting all packets with destination address equal to 127.0.0.0/8 is not equivalent to accepting `lo` either (packets directed to an address you own are routed to `lo` when you self connect)

Rules - 2

Matching Addresses and ports

- The most common match is the one checking either the source `-s` or the destination `-d` address
- It is possible to specify the mask as the number of contiguous ones `/n` or explicitly `/a.b.c.d`
- If the rule does not specify any mask, the default is /32, i.e. host only
- Also non contiguous masks are usable: e.g. 255.255.255.249 (0xFFFFFFF9) matches all the odd hosts up to .7
- Employing non contiguous masks may help in reducing the number of rules

Rules - 2

Matching Addresses

- The most common match is the one checking either the source `-s` or the destination `-d` address
- It is possible to specify the mask as the number of contiguous ones `/n` or explicitly `/a.b.c.d`
- If the rule does not specify any mask, the default is /32, i.e. host only
- Also non contiguous masks are usable: e.g. 255.255.255.249 (0xFFFFFFF9) matches all the odd hosts up to .7
- Employing non contiguous masks may help in reducing the number of rules

Rules - 3

Matching protocols

- After matching the address, the next most simple match is the one on the L4 protocol
- The `-p [tcp|udp|udplite|icmp|esp|ah|sctp|all]` option specifies the protocol to be matched
- Take care in not filtering fundamental ICMP messages, f.i. Type 3 (Destination Unreachable)
- Filtering non fundamental-but-useful messages (traceroute, echo/echo reply) is widely considered a brain damage unless specific reasons are in place

Rules - 4

Matching Ports

- The in addition to the source/destination address, also port matching is allowed via the `--sports/--dports`
- Both options allow to match a set of comma-separated ports (e.g. `--dport 22,80`)
- If the ports to be matched are contiguous, the range `:` operator can be used (e.g. `--dport 6881:6890`)
- The `--sports/--dports` need the `-p` option to be explicitly specified and to be matching either UDP or TCP

Rules - 5

Matching connection status

- The difference from a regular and a stateful packet filter resides in the ability to filter according to the connection status
- The `-m state --state <conn_state>` match allows to specify the status of the connection (for connection oriented protocols)
- Possible statuses are :
  - `NEW` : The packet beginning a connection (f.i.TCP/SYN)
  - `ESTABLISHED` : The packet is part of a connection flow
  - `RELATED` : The packet belongs to a related connection (f.i. active FTP mode)
  - `INVALID` : The packet does cannot be part of a valid connection (TCP SYN/FIN packets)
  - `UNTRACKED` : The packet is not being tracked

Rules - 6

Matching rate

- Sometimes it is desirable to limit the bandwidth for a specific class of connections
- The `-m limit <times/s>` match allows to send to the rule target only a specific amount of connections
- The `-m recent --set` option tags a connection as one of a set of recently happened ones
- The `-m recent --<time> <n> --hitcount` option allows to send to a target all the connections exceeding the hitcount/time
- Notice that rate limiting does not in any way limit the bandwidth of the single connection

Configuration Management

Saving and Restoring

- The `iptables` utility updates a rule at a time via Netlink
- In case multiple rule changes should be performed atomically it is not a good idea to call it a volley of times
- The `iptables-apply` is able to insert atomically the changes in the Netfilter tables
- The `iptables-save` and `iptables-restore` command provide a way of dumping and restoring a full ruleset at once
- There is also an `iptables-xml` utility which converts a ruleset in XML for whatever purposes it may have