VIRTUAL ACCESS

Service Provider Solutions

www.virtualaccess.com

Service Managed Gateway™

Using Advanced Filtering

Issue 1.0

Date 4 May 2010

1 About this document

1.1 Scope

This document describes:

- the principles of IP packet filtering;
- how to apply the filter to solve problems that you may encounter when dealing with a network router; and
- the full range of capabilities of IP filtering.

1.2 Readership

This document is for engineers who understand the fundamentals of routing, packet filtering and packet networking.

1.3 More information

Basic IP packet filtering concepts are described in the SMG full reference guide, 'IP System Configuration'.

2 Mandatory IP filter components

2.1 Filter types

Standard IP packet filters are categorised into seven actions, described in the table below.

Category	Description
PASS	Pass filters allow a packet to pass across the interface on which they are operating, if the packet meets the criteria specified in the rule. Packets not meeting the criteria specified in the filter are allowed to pass unhindered. Therefore, specifying PASS filters in isolation provides no filtering at all.
PASSIFKNOWN	Passifknown filters are similar to pass filters, but only allow inbound traffic through when there is a corresponding entry in the flow table for that interface. This usually implies that a corresponding outbound packet was transmitted across the same interface shortly beforehand. Packets that match the filter, but which are not present in the flow table will be blocked. If neither flow monitoring nor address translation is enabled on the interface, then this is treated as a normal pass filter.
PASSIFUP	Passifup filters are similar to pass filters, but can only be used on PPP interfaces configured for ISDN or V.90 connections. They implicitly specify the additional rule that the ISDN interface must be active, if a packet is to be allowed to pass. Thus if a packet is destined for the ISDN interface and such a filter rule is specified, then the packet will be allowed to pass, if it meets the criteria specified in the rule and the interface is already active. Should the packet meet the criteria specified in the rule and the ISDN interface is inactive, then the filter will prevent the packet from passing through. The ISDN call inactivity timer determines the active/inactive status of the ISDN interface. All packets passing across the ISDN interface will reset this timer, unless they pass through a PASSIFUP filter, in which case the ISDN interface status is interrogated first.
BLOCK	Block filters prevent a packet from passing across the interface on which they are operating, if the packet meets the criteria specified in the rule. Similar to PASS filters, packets not meeting the criteria specified in the filter are allowed to pass unhindered.
SUPPRESS **1-B CHANNEL** **2-B CHANNEL**	Suppress, 1-B channel and 2-B channel are bandwidth management filter types and are described in Bandwidth Management.

Table 1: Filter types and their descriptions

2.2 Interface to be filtered

Filters can operate on a single interface, a group of interfaces, or all interfaces.

When specifying filter rules the interface names used, such as. ppp-1 or eth-0, are not defined within the IP filter itself, but are part of the router's software. This means all the interface names available elsewhere in the router are available to the IP filter.

The special interface groups LAN and WAN are used to apply a filter to all interfaces configured as LAN or WAN types respectively. Configure this setting using the web setting 'Treat as WAN Interface', accessible under the advanced section of the IP configuration page for each device. For example, in the Expert View menu, click **Interfaces ->eth-x->ip ->ip.** Then, on the LAN Interface on eth-x page, click **Advanced**

By default, the private Ethernet interface eth-0 is configured as LAN while all other interfaces are WAN. Applying a filter to WAN interfaces is a convenient way of creating firewall-like rules to protect your network from Internet intruders, without affecting local access.

Where no interface is explicitly stated, the filter rule will be applied equally to all interfaces.

2.3 Filter direction

Filter rules are directional, in that they apply either to incoming (IN) or outgoing (OUT) packets. A filter cannot apply to both incoming and outgoing packets at the same time. If this type of functionality is required, two filter rules must be specified, one operating on the inward direction and the other operating on the outward direction.

2.4 Filter scope

The scope of the filter is defined in terms of the range of packet source and destination addresses that the filter applies to. At its widest, you can apply a filter to either all source and all destination addresses, or both. This will result in the filter rule applying to all packets that pass across the interface. You can also produce filters to apply to source or destination addresses from a particular sub-net, a single source or destination address or even a specific TCP or UDP port range.

2.5 Filter examples

```
1. pass in on PPP-1 from any to any
```

This filter allows packets containing any possible source address and any possible destination address, that is, from any to any, to pass across the interface PPP-1

Note: packets which do not satisfy the above criteria, will also be allowed to pass across the interface as this is the default action, that is, the action is taken when no rule matches a packet.

Warning! As this filter lets all packets pass it **you should not use it**. It only causes a performance hit on the throughput of the router. When specifying filters, be careful not to specify a redundancy!

```
2. block in on PPP-1 from any to any
```

This filter blocks all packets crossing the interface PPP-1, for the reasons explained in example 1. This filter is probably too heavy-handed for most applications, unless a number of PASS filters have been defined earlier to allow through specific packets.

```
3. block in on PPP-1 all
```

This filter has the same function as example 2. The only difference is that it uses the keyword 'all', which is a shorthand notation for 'from any to any'. The filter is identical, regardless of the syntax used to specify it.

```
4. passifup out on PPP-1 from any to any
```

This passifup filter allows packets from any address and destination for any address to pass, so long as the PPP-1 interface is active.

Note: filters themselves are strictly passive devices. While they can check the current status of an interface, they play no part in activating or deactivating that interface.

5. block in on eth-0 from 193.120.84.1 to any

This is another block filter. The difference between this and example 3 is that this filter only operates on packets originating from the IP address 193.120.84.1. The filter is not concerned about the destination of the packets, as can be seen by the use of 'to any'. A packet originating from a source with an IP address other than 193.120.84.1 will, by application of the default rule, be allowed to pass across the eth-0 interface.

6. block out on eth-0 from any to 193.120.84.1

This filter is similar to example 5. The difference being that packets destined for IP address 193.120.84.1 will be prevented from travelling across the eth-0 interface. The filter is not concerned with the origin of packets on the interface, as shown by the use of 'from any'. Again, applying the default rule, packets destined for an IP address other than 193.120.84.1 will be allowed to pass.

7. pass in on PPP-1 from 193.120.0.0/16 to any

This is a pass filter that operates on packets originating with devices on the 193.120.xxx.xxx subnet address range.

The expression 193.120.0.0/16 instructs the IP filter to build a mask using the first 16 bits of the specified IP address 193.120. All incoming packets will have the first two octets of their source address tested for equality with 193 and 120 respectively. The size of the mask is not restricted to 16 bits, you can apply any value between 0 and 32.

The final two zeroes in the expression 193.120.0.0/16 are not significant. Insert any value here. The resultant filter will be the same, as the 16-bit qualifier ensures that the final two octets in the address are masked out.

3 Optional IP filter components

The examples in section 2.5 adopt an all or nothing approach to the types of packet they filter. In practice, these filters are not very useful, as they do not allow packets passing across a network interface to be screened specifically.

It is unusual to want to simply prevent all inbound TCP packets from traversing an interface, but needing to block a subset of TCP packets is much more common.

This section introduces the more common parts of the filter used in real-world filtering applications. The optional IP filter components allow filters to be honed to only target specific types of IP packet.

3.1 Type of service

A type of service option allows an 8-bit service type value to be specified in a filter rule. The filter will check the specified value against that contained in an IP packet. The value must be specified in decimal.

The following filter example allows normal precedence packets requesting high reliability (R bit set) to pass across the interface eth-0.

```
pass in on eth-0 all tos 4
```

This example assumes the service type fields are set to zero. However, the filtering on this field is not reliable, as bits 6 and 7 of the service type field are unused and may not necessarily be set to zero.

3.2 Time to live (TTL)

This option permits filtering on the 8-bit 'time to live' field of the IP packet. The argument passed to TTL is measured in seconds. The filter rule attempts to match the TTL field in the packet with that specified in the TTL option.

In the following example, outgoing packets passing across the PPP-1 interface with a time to live field set at 30 seconds will be blocked:

```
block out on PPP-1 from any to any ttl 30
```

Routers will discard all packets whose TTL has been counted down to zero, so again, this option is not very useful.

3.3 Protocol

This option allows the IP filter to discriminate packets in terms of protocol of the encapsulated higher level packet. For example, an IP packet could be carrying a TCP packet.

Possible protocols that can be specified are 'TCP/UDP', 'TCP', 'UDP', 'ICMP', 'GRE', 'ESP' or 'AH'. If none of these are suitable, you can specify a protocol number.

Note: this option applies only to protocols carried by IP and not higher level application protocols. Therefore, do not use this option to explicitly block, for example, SNMP. When no protocol is specified, the filter applies equally to all protocols.

Note: Do not confuse protocol numbers with port numbers.

3.3.1 Protocol example 1

The following filters, used together, block all TCP or UDP packets from crossing the PPP-1 interface in either the inward or the outward directions.

```
1. block in on PPP-1 proto tcp/udp from any to any
```

```
2. block out on PPP-1 proto tcp/udp from any to any
```

3.3.2 Protocol example 2

The following filters, used together, allow only IP packets carrying ICMP to cross the eth-0 interface in the inward direction.

```
1. pass in on eth-0 proto icmp all
```

```
2. block out on eth-0 all
```

If an IP packet carrying an ICMP packet enters the router on interface eth-0, it will match the first rule and so the IP filter will return 'pass'. If the IP packet is not carrying an ICMP packet then the IP filter will fall through to the second rule, resulting in the packet being blocked.

3.3.3 Protocol example 3

This filter will block the IGMP (Internet Group Multicast Protocol) from entering on the PPP-1 interface. IGMP has a protocol number 3.

```
block in on PPP-1 all proto 3
```

3.4 Address masking

A filter can operate on segments of an IP address without necessarily looking at the whole address. The following address masking examples show ways in which this can be done.

3.4.1 Address masking example 1

To block incoming TCP packets on the interface PPP-1, originating from all class C IP sub-networks whose addresses start with the prefix 201.113.71 you can use any of the following rules.

```
1. block in on PPP-1 from 201.113.71.0/24 to any
```

Note: the /24 masks the most significant 24-bits of the address, that is, 201.113.71.

Other ways of specifying the same rule are given in the next 2 examples.

```
2. block in on PPP-1 from 201.113.71.0/255.255.255.0 to any
```

or:

```
3. block in on PPP-1 from 201.113.71.0 mask 255.255.255.0 to
any
```

Examples 2 and 3 are more intuitive than example 1.

3.4.2 Address masking example 2

The mask can be specified in hexadecimal notation as in the example below.

```
block in on PPP-1 from 201.113.71.0 mask 0xffffff00 to any
```

Masks are always in use whether specified or not. The default mask is /32 or 255.255.255.255.

3.5 Ports

One of the more useful options is filtering on the basis of the port numbers carried in TCP or UDP packets. Unlike other numeric values that can be specified in filter rules, you can compare ports with a prescribed value. You can test both the source and destination ports depending on the point in the rule that the test is inserted.

When testing ports the following single argument operations can be applied:

Operator	Alias	Parameters	Meaning
<	Lt	port number	Port in packet is less than value supplied.
>	Gt	port number	Port in packet is greater than value supplied.
=	Eq	port number	Port in packet is equal to value supplied.
!=	Ne	port number	Port in packet is not equal to value supplied.
<=	Le	port number	Port in packet is less than or equal to value supplied.
>=	Ge	port number	Port in packet is greater than or equal to value supplied.

Note: the operator or its alias may be used when specifying filter rules.

3.5.1 Ports example 1

To permit only FTP (port 21) and Telnet (port 23) traffic to enter the router on interface eth-0, use the following filter.

```
1. pass in on eth-0 proto tcp from any to any port = 21
```

```
2. pass in on eth-0 proto tcp from any to any port = 23
```

```
3. block in on eth-0 all
```

Note: when testing port numbers, a protocol of TCP/UDP is assumed unless explicitly specified.

Because the port test has been specified after the second 'any' in the rule, that is, the destination address; the IP filter assumes that the required port is the destination port. Putting the port test after the first 'any' will result in the source address being tested. See Ports example 2.

3.5.2 Ports example 2

To block packets with source ports greater than, or equal to 1024 entering the router on interface eth-0, use the following filter.

```
block in on eth-0 proto tcp/udp from any port GE 1024 to any
```

3.5.3 Ports example 3

To block incoming SNMP traffic on interface PPP-1, use the following filter.

```
block in on PPP-1 proto udp from any to any port = 161
```

There are also two range operators, defined in the table below.

Operator	Parameters	Meaning
<>	Two port numbers, e.g. p <> p	Port in packet (p_p) is in the following range: p <= p<=p
> <	Two port numbers, e.g. p >< p	Port in packet (p_p) is in the following range: p < p OR p >p

3.5.4 Ports example 4

To block incoming UDP packets on interface PPP-1 with destination ports in the range 3000 to 3010, use the following filter.

```
block in on PPP-1 proto udp from any to any port 3000 <> 3010
```

3.5.5 Ports example 5

To permit only incoming UDP packets on interface PPP-1 with destination ports in the range 3000 to 3010, use the following filter.

```
block in on PPP-1 proto udp from any to any port 3000 >< 3010
```

3.6 TCP packet flags

It is possible to filter on the flags carried within the code bits field of the TCP packet. As well as being able to filter on the presence of certain flags. It is also possible to filter on the absence of flags. Each of the defined TCP flags is denoted by the letters shown below:

Acronym	Flag	Description
S	SYN	Synchronize
A	ACK	Acknowledgement
P	PSH	Push
U	URG	Urgent
R	RST	Reset
F	FIN	Fini

Table 2: Denotations of TCP flags and their descriptions

3.6.1 Syntax of the flag option

Before using the TCP packet flags as filters, it is necessary to understand the syntax of the 'flags' option, shown in the example below.

```
flags flag-list "/" flag-mask
```

The flag-mask specifies the list of flags to consider before deciding whether the packet in question matches the filter rule. The flag-list lists the flags to set in the packet, for the packet to match the filter rule. The flags are specified by their acronym.

Note: To test for a flag NOT being set, the flag in question should be put into the flag-mask and left out of the flag-list.

3.6.2 TCP packet flags example 1

```
block in on PPP-1 proto tcp from any to any flags S/S
```

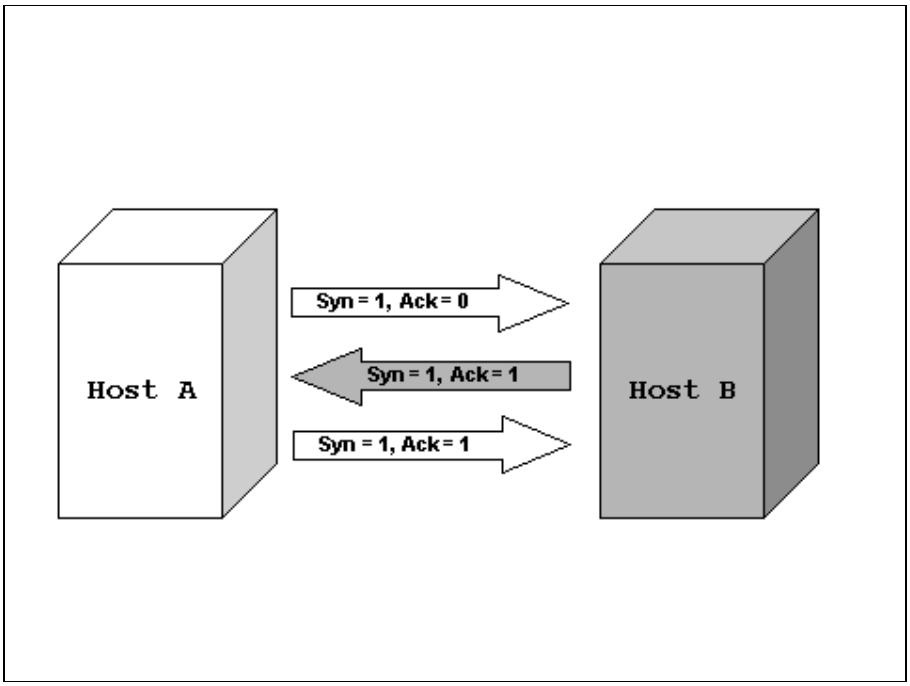
The flag-mask "/S" indicates that the only flag relevant for filtering is the SYN flag. The flag-list "S" also specifies only the SYN flag. The filter blocks all incoming TCP packets on interface PPP-1 whose SYN flag is set. This is true regardless of the status of any of the other flags. The only flag that the filter tests, is the status of the SYN flag.

3.6.3 TCP packet flags example 2

```
block in on PPP-1 proto tcp from any to any flags S/SA
```

This time the flag-mask specifies the SYN "S" and ACK "/SA" flags. However, the flag-list still only specifies the SYN flag "S". This filter will match only incoming TCP packets on interface PPP-1 whose SYN flag is set and whose ACK flag is clear, that is, not set.

Filtering on the SYN and ACK flags is particularly useful during connection setup. Consider two hosts: A and B, establishing a TCP session.



1. Host A sends a TCP packet with SYN set.
2. Host B responds with a packet with SYN and ACK set
3. Host A then replies with a packet with ACK set.

If host A were using the rule in TCP packet flags example 1, it would be able to connect to host B but host B would not be able to connect to A. If host A were using the rule in

TCP packet flags example 2, then neither connection direction would work because the second part of the handshake would be dropped.

Being able to check other combinations of flags is sometimes useful too: for example, the combination of SYN and FIN is sometimes used to probe for IMAP (Internet Message Access Protocol) servers.

3.7 IP packet options

IP packets can be filtered based on the options that the packet carries. You can filter packets only for the presence or absence of options. You can also test for the presence or the absence of specific options. You can filter for the options listed below.

addext	cipso	encode	eip	e-sec
finn	imitd	lsrr	mtup	Mtur
nop	rr	satid	sec	sec-class
ssrr	ts	tr	visa	zsu

Note: The sec-class option is more complicated than the other options. It is described in more detail in IP Security Classes .

3.7.1 IP packet options example 1

To block packets carrying IP options from crossing the eth-0 interface, you can use the following filter.

```
block in on eth-0 from any to any with ipopt
```

3.7.2 IP packet options example 2

To block IP packets that are not carrying any IP options, you can use the following filter.

```
block in on eth-0 from any to any with no ipopt
```

Note: you can use "Not" in the place of "no" in the above filter.

3.7.3 IP packet options example 3

This filter will block IP packets carrying the options RR, zsu and encode crossing the eth-0 interface:

```
block in on eth-0 from any to any with opt RR,zsu,encode
```

3.8 ICMP message types and codes

You can selectively filter ICMP packets depending on the type of message carried within the packet. The table below describes the range of message types that can be blocked.

Type field	ICMP message type
echorep	Echo Reply
unreach	Destination Unreachable
squench	Source Quench
redir	Redirect
echo	Echo Request

timex	Time Exceeded (for a datagram)
paramprob	Parameter Problem on a Datagram
timest	Timestamp Request
timestrep	Timestamp Reply
inforeq	Information Request (obsolete)
inforep	Information Reply (obsolete)
maskreq	Address Mask Request
maskrep	Address Mask Reply

3.8.1 ICMP message type example 1

Use this filter to block address mask request packets crossing the PPP-1 interface.

```
block in on PPP-1 proto icmp from any to any icmp-type maskreq
```

3.8.2 ICMP message type example 2

If required, you can block all ICMP packets from crossing the interface PPP-1 using this filter.

```
block in on PPP-1 proto icmp all
```

The disadvantage of this filter is that it blocks some of the more useful ICMP messages such as echo, which is used by the ping command.

As well as filtering on the ICMP message type field, it is also possible to filter on the ICMP code field. The main user of the ICMP packet's code field is the destination unreachable message.

The IP filter supports the following code values for use with the destination unreachable message.

Code Mnemonic	value
net-unr	0
host-unr	1
proto-unr	2
port-unr	3
Needfrag	4
Srcfail	5
net-unk	6
host-unk	7
Isolate	8
net-prohib	9
host-prohib	10
net-tos	11
host-tos	12

For example to block ICMP destination unreachable messages carrying the port unreachable code, use the following filter.

```
block in on PPP-1 proto icmp from any to any icmp-type unreach
code port-unr
```

Note: numbers can also be used in conjunction with the code field. For example, the above filter could be re-written as:

```
block in on PPP-1 proto icmp from any to any icmp-type unreach
code 3
```

3.9 IP security classes

Filtering for IP packets that carry a given security class is a special case of filtering general IP packet options. For more information, read IP Packet Options in the SMG full reference guide.

The list below shows security classes that you can filter.

reserv-4	confid	unclass	reserv-2
topsecret	secret	reserv-3	reserv-1

3.9.1 IP security classes example 1

To block IP packets with security class reserv-3 crossing the interface PPP-1, use the following filter.

```
block in on PPP-1 all with opt sec-class reserv-3
```

3.9.2 IP security classes example 2

To permit all IP packets that do not have the 'topsecret' security option set to cross interface eth-0, use the following filter.

```
pass in on eth-0 from ant to any with not opt sec-class
topsecret
```

3.10 Time-sensitive filtering

Filters can be made time-sensitive as they are only active during certain times of the day or certain days of the week. This is a useful facility for removing certain streams that may be useful or even required at other times of the day, for example, disabling WWW access during working hours.

Specify switch-on and switch-off times using the 24-hour clock.

3.10.1 Time-sensitive example 1

```
block in on PPP-1 proto tcp from all between 08:00 - 18:00
```

This filter blocks all incoming and outgoing TCP and UDP traffic across interface PPP-1 between the hours of 8 AM and 6 PM. As it stands the filter is not particularly useful, as it blocks too broad a potential range of traffic.

This time range is consistent across all seven days of the week, that is, the filter will operate between 8 AM and 6 PM, Sunday through Monday.

3.10.2 Time-sensitive example 2

The addition of the day of the week in the filter in example 2 changes the filter such that it is only active on Mondays.

```
block in on PPP-1 proto tcp from all between mon 08:00 - 18:00
```

3.10.3 Time-sensitive example 3

To extend the range of operation of the filter to all weekdays, configure the filter as shown below.

```
block in on PPP-1 proto tcp from all between mon - fri 08:00 -
18:00
```

The range of operation of this filter is 8 AM to 6 PM on each weekday, that is, Monday through Friday.

3.10.4 Time-sensitive example 4

```
block in on PPP-1 proto tcp from all between fri - mon 08:00 -
18:00
```

This filter is operational between the hours of 8 AM and 6 PM on Friday, Saturday, Sunday and Monday.

3.10.5 Time-sensitive example 5

```
block in on PPP-1 proto tcp from all between mon 08:00 - fri
18:00
```

This filter is effective from 8 AM on Monday morning through 6 PM on Friday.

3.10.6 Time-sensitive example 6

This filter will block outgoing traffic on interface *eth-0* to any server listening on TCP port 8080, a WWW server, in most cases. Traffic will be blocked during working hours - 9am to 5:30pm - and permitted at other times including weekends.

```
block out on eth-0 proto tcp from any to any port = 8080
between mon - fri 09:00 - 17:30
```

3.11 The NULL filter

The null filter does nothing. It only acts as a placeholder for other filters and its only practical use is in the embedded web interface provided for filter construction.

3.11.1 Null filter example 1

```
null
```

When processing a list of filters, the IP filter will ignore any null filters and move on to the next non-null filter.

4 IP filter syntax

This section explains the steps for constructing a filter. Optional syntax is indicated by square brackets []; variable values are indicated in italics.

1. First, use one of the following filter actions:

```
block [ return-icmp {"(" <number> ")"} | return-rst ]
count
pass
passifknown
passifup
log [ body | first ]
null
```

If null has been chosen in the previous step, then the filter is complete. Otherwise, continue on with one of the following lines.

```
in | out [ log [body | first | or-block ] ]
```

2. Followed by:

```
[on <interface> [dup-to <interface >] [([to <interface>])|fastroute]]
```

where <interface> is a character string.

3. Next, specify the type of service.

```
[ tos <service-type> ]
```

where <service-type> is a positive integer

4. The time to live:

```
[ ttl <number> ]
```

5. The protocol

```
[proto tcp/udp | tcp | udp | icmp | <number>]
```

6. Then, one of the following:

```
all
from hostname | a.b.c.d | any [/ (a.b.c.d | <number>) | mask (a.b.c.d |
a.b.c | a.b | a) ] [<port-exp>]
```

where hostname is a name that can be found in /etc/hosts or DNS lookup, and a.b.c.d is a dotted IP address, and:

```
<port-exp> == PORT = | eq | != | n e | < | lt | > | gt | <= | le | >= |
GE<number>
```

or

```
<port-exp> == PORT a <> b
```

This means port < a or port > b, that is, port outside range a...b.

or

```
<port-exp> == PORT a >< b
```

This means a _ port _ b, that is, port inside range a...b

7. Next, type in the optional keywords.

```
{to ...}
```

'to' has exactly the same options as 'from'.

8. After the optional keywords, enter any flag filters.

```
[ flags <set of flags> [/ <set of flags> ] ]
```

where <set of flags> includes one of more of the following: F, S, R, P, A, U

9. Type in the following parameters.

```
[ with | and) [(no | not)] [ipopt | opt <opt-name> {, <opt-name>},[sec-
class <sec-class-name> {, <sec-class-name>}] ]
```

where opt-name == nop | RR | zsu | mtup | mtur | encode | Ts | tr | sec | sec-class | lsrr | e-sec | cipso | satid | ssrr | addext | visa | imitd | eip | Finn

where sec-class-name == reserv-4 | topsecret | secret | reserv-3 | confid | unclass | reserv-2 | reserv-1

10. If you require ICMP packet type filtering, type in the following

```
[ icmp-type [[ <icmp-type-string>] [ code <number>] ]
```

where icmp-type-string == echorep | unreach | squench | redir | echo | Timex | paramprob | timest | timestrep | inforeq | inforep | maskreq | maskrep

11. If the filter needs to be time sensitive:

```
between (time – time) | (day time – [day] time) | (day – day time –
time)
```

where:

time == hour : minute

day == sun | mon | tue | wed | thu | fri | sat

hour == 1 | 01 | 2 | 02 | ... | 23 -- i.e. 24 hour clock

minute == 01 | 02 | ... | 59

4.1 Notes

- IN is not permitted if you have previously specified RETURN-RST or RETURN-ICMP.
- Can only be used with PASS filters.
- If RETURN-RST has been specified above then TCP is the only permissible option here.
- This is equivalent to "FROM ANY TO ANY". Note that "PORT" cannot be used with all
- If ANY is used, then applying a bit-mask (e.g. "/", "MASK") yields unpredictable (i.e. incorrect) results.
- The number refers to the most significant bits, the remaining bits are masked out, e.g. 193.120.113.75/16 _ 193.120.0.0
- A.B.C _ A.B.C.0, A.B _ A.B.0.0 - although this later case appears to be interpreted as A.0.0.B! Also A on its own is interpreted as 0.0.0.A.
- If "FROM" has been used then "TO" is mandatory.
- When OPT is used, the function that reconstructs the filter string appears to include the protocol twice.
- Do not precede SEC-CLASS with white space or the parser will reject the filter.
- Can only be used if filtering on ICMP protocol.