

LOGARITHMIC TIME RANGE-BASED MULTIFIELD-CORRELATION PACKET CLASSIFICATION

TECHNICAL FIELD

5 This disclosure relates to network packet classification by a logarithmic time range-based multifield-correlation packet classifier.

BACKGROUND

Network devices may be used to perform actions on the
10 packets of data flowing through them. The actions performed may vary with the kind of data traffic flowing through the device and on the policies set up at the device. For example, the device can have a policy for marking all traffic from a particular subnet with high priority, send
15 specific flows through a particular MPLS (Multi-protocol Label Switch) tunnel or VPN (Virtual Private Network), or collect per-flow statistics for accounting/billing purposes.

Packet classifiers are used to determine the policy applied by the network device. The policies applied to the
20 packet may be determined, for example, from a search of the fields in the packet header. The packet classifiers (also known as "filters") for the policies may be searched for matches with the packet header field each time a packet arrives at the device.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is an example of a network for transmitting and receiving data packets.

FIG. 2 is an illustrative table of policies and associated data-packet header field filters.

FIG 3 is a source address filter tree diagram for the table of FIG. 2.

FIG 4 is a destination address filter tree diagram for the table of FIG. 2.

FIG 5 is a source-port-filter tree diagram for the table of FIG. 2.

FIG. 6 is a flow chart of a process to determine the policies associated with a packet header field.

FIG. 7 is a flow chart of a process for determining the data-packet classification.

FIG. 8 is a table illustrating filter-identifiers.

FIG. 9 illustrates the table of FIG. 8 sorted by filter-ID;

FIG. 10 illustrates the table of FIG. 8 sorted by data-packet classifier field entries.

FIG. 11 illustrates the table of FIG. 10 sorted by filter-ID.

FIG. 12 is a flow chart of a process to associate two policy-ID sets with each node of a filter tree.

FIG. 13 is a source-port filter tree diagram for the table of FIG. 2 having two policy-ID sets associated with each tree node.

DETAILED DESCRIPTION

FIG. 1 illustrates a network 10 that may be employed to relay packets of information between nodes 12, 14 and 16. Although only three nodes are shown, the network 10 may

couple many nodes. Nodes 12, 14 and 16 include network devices 18, 20 and 22, respectively. Node 16 illustrates, by way of example, that a node also may include a processor 24 and memory 26. Network device 22 may be used in
5 conjunction with the processor 24 and memory 26 to provide network services on the packets of information received at node 16. Data packets may be transferred between nodes on the network.

A data-packet received by a node may include a data-
10 packet header field to indicate one or more characteristics of the data. For example, the header may indicate an address from where the data-packet was sent, an address where the data-packet is to be sent, an operation to be performed on all or part of the data, etc.

15 Network nodes have policies associated with them. Policy-based networking is the management of a network so that various kinds of traffic - data, voice and video, for example - receive the priority of availability and bandwidth needed to serve the network's users effectively.

20 The policies may be stored in a policy repository such as memory 26. The network device can communicate the data-packet header field entries to a processor 24. Processor 24 operates on the entries to classify the data-packet in terms of the policies that are applicable to the data-packet. The
25 applicable policies may be retrieved by the processor from

the policy repository in memory 26.

The policies that are applicable to a particular data-packet may be determined from the entries in the data-packet header fields. A policy may be applicable, in a particular
5 example, to all data-packets originating from a certain source address. Other policies may be range-based. "Range-based" policies are policies that apply to data-packets that have header field entries that fall within a range of values. For example, a policy may apply to all data-packets
10 originating from a source having an address within a range of addresses. The policies, in the particular example, have a source address or range of addresses to filter the data-packets and determine to which packets a policy applies.

One implementation of the packet classification employs
15 a red-black balanced binary tree created for each filter element type corresponding to a particular data-packet header field. FIG. 2 illustrates a particular implementation having a set of policies, policy-ID (policy-identifier) 202, associated with a 3-tuple packet
20 classification of source address 204, destination address 206 and source port 208. FIGS. 3, 4 and 5 illustrate, in tree form, the source address, destination address and source port filters, respectively, of the policies illustrated in FIG. 2. The number of packet header fields
25 employed in the classification is not limited to the

specific fields illustrated.

As an example of the tree construction, FIG. 3 illustrates a source-address tree 300 of the source addresses 204 associated with the policy-IDs 202 of FIG. 2. Each of the filter nodes, 310-320, is an address from the source addresses 204 of FIG. 2. Filter nodes 310-320 are arranged in order of their value. In the tree illustrated, the filter nodes are arranged in ascending order of their address value from left to right. However, the filter nodes may be arranged in other hierarchical orders. The policies associated with each of the filter nodes 310-320 are illustrated in reference policy boxes 322-332.

Each of the reference policy boxes 322-332 holds a reference to the list of policies that match the filter node. For example, referring to FIG. 2, a packet-header source address of 7.8.9.10 falls within the range of source addresses that implicate the policies identified by policy-IDs P1, P2, P3 and P5. These policy-IDs are illustrated in FIG. 3 reference policy box 326. The remaining reference policy boxes 322, 324 and 328-332 are similarly determined. The reference policy boxes 410-420 for the destination-address tree 400 of FIG. 4 and the reference policy boxes 510-516 for the source-port tree 500 of FIG. 5 may be determined in an analogous manner.

The policy having the policy-ID of P3 in FIG. 2 is

applied to packets that have the following entries in the header fields: (1) a Source Address in the range of addresses from 1.2.3.4 to 10.10.10.10, (2) a Destination Address in the range of addresses from 11.12.13.10 to 11.12.13.50, and (3) a Source Port of 3000.

FIG. 6 illustrates a process to determine the policies associated with a packet header. The process may be implemented as a program executable by a processor, or using other circuitry. A tree is selected at 602. The selected tree is searched for the value matching the value in the packet header at 604. If a filter node having a value matching the value in the packet header is found at 606, the reference policies associated with the found filter node are ascertained at 608 and the program proceeds to 614.

If a match for the packet header value is not found at 606, both the filter node with a value just less and the filter node with a value just greater than the packet header value are selected at 610. A non-existent node to either higher or lower is considered a null set. The list of matches for this packet header value is the set intersection of the list of policies associated with these filter nodes and is determined at 612.

Blocks 602-612 are repeated until the policy lists from each tree are obtained at 614. A set intersection of all the obtained lists provides the set of policies that match the

data packet at 616.

In an illustrative example, the trees of FIGS. 3-5 can be used for determining the set of policies that match a packet having a source address 5.6.7.8, destination address

5 11.12.13.40 and source port 3000.

Referring to FIG. 3, the source-address tree has a filter node 312 that is a match for the packet-header source address 5.6.7.8. The set of policies obtained is {P1, P3, P5} as illustrated in reference policy box 324.

10 Referring to FIG. 4, the destination-address tree does not have a filter node that is a match for the packet-header destination address 11.12.13.40. The filter node having a value just below this address is filter node 406 which has the value 11.12.13.10. The set of policies obtained for filter
15 node 406 is {P1, P3} as illustrated in reference policy box 414. The filter node having a value just above the packet-header destination address is filter node 408 which has the value 11.12.13.50. The set of policies obtained for filter node 408 is {P3} as illustrated in reference policy box 416.
20 The intersection of the two obtained policy sets is:

$$\{P1, P3\} \cap \{P3\} = \{P3\}$$

This is the set of policies associated with the destination address 11.12.13.40 of this example.

Referring to FIG. 5, the source-port tree has a filter
25 node 508 that is a match for the packet header source port

address 3000. The set of policies obtained is {P3, P4, P5} as illustrated in reference policy box 514.

The set intersection of policies obtained from all of the trees:

$$5 \quad \{P1, P3, P5\} \cap \{P3\} \cap \{P3, P4, P5\} = \{P3\}$$

The result is {P3} as the policy-ID match for this packet.

In another implementation, FIG. 12 illustrates a process 800 for creating a tree where the intersection sets
10 for the trees may be computed when the policies are installed in the tree rather than at run-time, when the packet arrives. In this implementation, two sets of policy-IDs may be associated with each node value of a tree as described below. A table of policies associated with a
15 packet-header classifier-field is selected at 802. A filter node for each policy endpoint is entered into the tree at 804. A first set of policy-IDs is associated with each node at 806. The first set is a set of policy-IDs from the table that may be applicable to packets having packet-header
20 classifier-field values exactly matching the node value. The first set may be called the exact-match set. A second set of policy-IDs is associated with each node at 808. The second set includes the policy-IDs from the table that may be applied to packets having packet-header classifier-field
25 values between the node value and the next higher node value

in the tree. The second set may be called the range-based set. This association of two sets of policy-IDs may be accomplished for each packet-header classifier-field.

FIG. 13 illustrates a source-port filter node tree 900 of the policy table of FIG. 2 having an exact-match policy-ID set and a range-based policy-ID set associated with each filter node. Each of the filter nodes, 902-908, is an address from the source port of FIG. 2. Filter nodes 902-908 are arranged in order of their value. In the tree illustrated, the filter nodes are arranged in ascending order of their value from left to right. However, the filter nodes may be arranged in other hierarchical orders. The policies associated with each of the filter nodes 902-908 are illustrated in reference policy boxes 910-916.

Each node is associated with two policy-ID sets: an exact-match policy-ID set 910-916 and a range-based policy-ID set 918-924. The exact-match policy-ID set is a set of policies that match the filter node. For example, referring to FIG. 2, a packet-header source port of 1000 falls within the range of source addresses that implicate the policies identified by policy-IDs P1, P2 and P5. These policy-IDs are illustrated in FIG. 13 exact-match set 910. The remaining exact-match sets 912, 914 and 916 are similarly determined.

The exact-match set associated with a node is applied

to a packet having a packet-header source port value exactly matching the node value.

The range-based policy-ID set is a set of policies that may be applied to a packet having a packet-header classifier-
5 field value that is not an exact match for a node value. The range-based set is the set intersection between a node and the next higher value node. For example, range-based set 918 is associated with node 902 having the exact-match set $910 = \{P1, P2, P5\}$. The next higher node is 904 having the exact-match
10 set $912 = \{P2, P4, P5\}$. The intersection of the two obtained policy sets is:

$$\{P1, P2, P5\} \cap \{P2, P4, P5\} = \{P2, P5\}$$

The set $\{P2, P5\}$ is the range-based set 918 associated with node 902.

15 An exact-match set of a non-existing node is defined as the null set. Node 908 has an exact-match set $\{P5\}$. Because there is no next higher node after node 908, the next higher node exact-match set is the null set. The intersection set is:

20 $\{P5\} \cap \{\text{null}\} = \{\text{null}\}$

The null set $\{\text{null}\}$ is the range-based set 924 associated with node 908.

Range-based nodes 920, 922 associated with nodes 904, 906, respectively, may determined in a manner similar to that
25 described above.

Some implementations computing the range-based set before run-time may have an advantage of improving the performance of the search method at run-time because the intersection sets have already been calculated. Another
5 advantage of some implementations is that the intersection set need not be recomputed for successive arrivals of packets having packet-header classifier-fields having values that are not exact matches for a node value in a tree.

As an example, assume that a packet has a packet-header
10 source field of 2000. Referring to FIG. 13, the source port value has an exact-match with the value associated with node 902. The exact-match set of policy-IDs 912 is {P2, P4, P5}. If the source port value is 2500, there is no node in the tree having an exact match for that value. The range-based
15 policy-IDs of the set associated with the node value just below the packet value are applicable to the packet. In this example, node 904 with a value of 2000 is the node just below the packet-header source port value of 2500. The policies relevant to this example packet is the range-based
20 set {P4, P5}.

In a similar manner the policies relevant to other packet-header classifier-fields of the packet may be determined from trees having an exact-match policy-ID set and a range-based policy-ID set associated with each node.

25 The method associates filter-IDs with policies. The

filter-IDs can provide an opportunity to group the policies in a logical manner. The method may identify the policies applicable to a packet by grouping the filter-IDs into non-intersecting sets and ordering the
5 sets for each filter element using another balanced binary tree. Using this method, it is possible to shorten the time to determine the set intersection to approach $O(\log n)$ for each filter element type searched. This may result in a searching time approximately $[O(\log$
10 $n)$ lookup per matching filter-ID] * [the number of filter element types searched].

The filter-IDs are logically divided to remove the randomness that could cause $O((\log n)*n)$ worst case performance. Logically non-applicable filter-IDs may be
15 skipped using the technique described below in association with FIGS. 8 and 9.

In one implementation, the filter-ID may be preceded by a bit mask descriptor. Each bit in the bit mask represents a particular filter-type element. If the bit
20 is set, then that particular filter-type element represents a range. If the bit is not set, then the filter element is an exact value (not a multi-value range).

Referring to FIG. 2, suppose the filter-ID has eight
25 bits with the highest order three bits representing the

bit mask for the three filter elements (source address, destination address and source port, in that order).

Then filter-IDs for the policies P1 to P5 may be determined as:

5

P1: 1 0 0 0 0 0 0 1

P2: 0 0 1 0 0 0 1 0

P3: 1 1 0 0 0 0 1 1

P4: 0 0 1 0 0 1 0 0

10

P5: 1 1 1 0 0 1 0 1

For policy P1, the illustrated case has the lower 5 bits identifying P1 as 00001. In this case 5 bits were allocated for a policy number although this is not a limitation of the disclosure. The highest order 3 bits are the bit mask. The highest order bit is set to 1 to indicate that the source address filter for P1 is range-based. The next two bits are set to 0 to indicate that the destination address and source port filters are exact values. The combination of the binary bit mask 100 and P1 identifier of 00001 results in a filter-ID of decimal 129. Policies P2-P5 are determined in a similar manner for this example. The table may be sorted by filter-ID, logically grouping the filter-IDs based on the bit mask.

25

FIG. 7 illustrates a method of computing the set

intersection from the sets obtained for each data-packet header field. At 710, a filter-ID may be associated with each policy-ID in a policy-ID table for each packet header field as described above. The policy tables may
5 then be sorted by filter-ID at 712.

A set of filter-IDs applicable to each data-packet header field may be obtained maintaining the sorted order at 714. At 718 the lowest filter-ID is selected as a search-ID from a set of filter-IDs that is comprised of
10 the largest filter-ID from each data-packet header set. The remaining data-packet sets are searched for a match for the search-ID at 720.

If a match for the search-ID is found at 722 the filter-ID corresponding to the search-ID is added to the
15 result intersection set at 724. A test 722 is performed for a match of the search-ID. If the search-ID is not found at 722, a test 726 is performed to determine if the end of any data-packet set has been reached. If the end of a data-packet set is not determined at 726, the next
20 largest filter-ID is selected that does not exceed the search-ID to be a new search-ID at 730. The processor then loops back to 720 where the other data-packet sets are again searched for a match. Blocks 720-730 are repeated until the end of a data-packet set is determined
25 at 726 and the process stops at 728.

As an illustrative example, FIG. 8 is a policy table having two filter elements, destination port and source port, and twelve policy entries. A bit mask may be prefixed to a policy number as described above to yield a
5 bit representation and resultant filter-ID for each policy in the table. The table may then be sorted by filter-ID to yield the table of FIG. 9.

In another illustrative example, it is assumed that a packet having a source port entry of 4300, destination
10 port entry of 1300 and source address of 8300 arrives at the network device. Referring to FIG. 9, the set of policy-IDs and associated filter-IDs obtained for each packet header field addresses is:

15 Source Port : {P7, P8, P9, P10, P11, P12, P4} = {71, 72, 73, 74, 75, 76, 132}

Destination Port: {P9, P1, P2, P3, P4, P5, P6} =
{73, 129, 130, 131, 132, 133, 134}

Source Address: {P8, P9, P11, P1, P4, P6} = {72, 73,
20 75, 129, 132, 134}

The set of largest filter-IDs obtained from each set is {source port, destination port, source address} =
{132, 134, 134}. The lowest of these filter-IDs, 132, is
25 selected as the search-ID.

The remaining sets are searched for a filter-ID that matches the search-ID. In the illustrated example, the destination port and source address sets are searched from the highest towards the lowest filter-ID. The search-ID, 132, is found in all the remaining sets and is added to the result set intersection.

Because the method has not exhausted the filter-IDs in any set, the next largest filter-ID that does not exceed the search-ID is selected from the sets. This filter-ID is used as the new search-ID. In the illustrated example, the filter-IDs that have not been searched in the sets are source port = {71, 72, 73, 74, 75, 76}, destination port = {73, 129, 130, 131} and source address = {72, 73, 75, 129}. The set of largest filter-IDs is {76, 131, 129}. The lowest of these, filter-ID = 76, is selected as the new search-ID.

The remaining sets are searched for a filter-ID that matches the new search-ID. The destination port and source address sets of filter-IDs below the previous search-ID are searched towards the lowest filter-ID for a match to the new search-ID. In this example, the search-ID, 76, is not found in all the remaining sets and is not added to the result set intersection.

Because the method still has not exhausted the filter-IDs in any set, the next largest filter-ID that

does not exceed the search-ID is selected from the sets.

This filter-ID is used as the new search-ID. In the illustrated example, the filter-IDs that have not been searched in the sets are source port = {71, 72, 73, 74, 75}, destination port = {73} and source address = {72, 73}. The set of largest filter-IDs is {75, 73, 73}. The lowest of these, filter-ID = 73, is selected as the new search-ID.

The remaining sets are searched for a filter-ID that matches the search-ID. In the illustrated example, the destination port and source port sets are searched from the highest towards the lowest filter-ID. In this example, the search-ID, 73, is found in the all the remaining sets and is added to the result intersection set.

The method has reached the lowest value filter-ID of the source port set and the method is completed. The result set intersection is {P4(132), P9(73)}.

In another implementation, a hierarchical order of the data-packet classifier fields may be used as a filter-ID to order the filters. For example, a filter specifying a source port field entry of 10, destination port field entry of 20 and source address of 30 may have a filter-ID of (10, 20, 30). In this example, the order of significance is, for sorting purposes, source port

field entry, destination port field entry and then source address field entry. The filter-ID may also be derived from the packet field entries. As an example, the filter-ID may be one or more of the classifier field values divided by a constant, or operated on in any manner that produces a logical sequencing of the filter-IDs. The sorting characteristics of the filter packet fields are not limited to the number, type, and hierarchical order of the filter-IDs.

FIG. 10 illustrates the filter-IDs derived from the filter table of FIG. 8 using the source port field entry, destination port field entry and source address field entry, as described above, as the filter-ID.

FIG. 11 is the policy table of FIG. 10 sorted by filter-ID, from lowest to highest. The order of policies is determined by comparing the source port field, then the destination port field and then the source address field of the filter ID to determine the hierarchical order of the filter-IDs.

Continuing with the illustrative example, it is assumed that a packet having a source port entry of 4300, destination port entry of 1300 and source address entry of 8300 arrives at the network device. Referring to FIG. 11, the set obtained of policy-IDs and associated filter-IDs for each packet header field addresses is:

Source port: {P7(4000 , 1100, 8100), P8(4000, 1200,
8300), P9(4000, 1300, 8300), P10(4000 ,1400, 8400),
P11(4000, 1500, 8300), P12(4000, 2000, 9500), P4(4300,
5 1000, 8300)}

Destination port: {{P1(4000, 1000, 8300), P9(4000,
1300, 8300), P2(4100, 1000, 8100), P3(4200, 1000, 8200),
P4(4300, 1000, 8300), P5(4400, 1000, 8400), P6(4500,
10 1000, 8300)}

Source address: {{P1(4000, 1000, 8300), P8(4000,
1200, 8300), P9(4000, 1300, 8300), P11(4000, 1500, 8300),
P4(4300, 1000, 8300), P6(4500, 1000, 8300)}

15

The set of largest filter-IDs from each set obtained
is {P4(4300, 1000, 8300), P6(4500, 1000, 8300), P6(4500,
1000, 8300)}. The lowest of these filter-IDs, P4(4300,
1000, 8300), is selected as the search-ID.

20 The remaining sets are searched for a filter-ID that
matches the search-ID. In the illustrated example, the
destination port and the source address sets are searched
from the highest towards the lowest filter-ID. The
search-ID, (4300, 1000, 8300), is found in all the
25 remaining sets and is added to the result intersection

set.

Because the method has not yet exhausted the filter-IDs in any set, the next largest filter-ID that does not exceed the search-ID is selected from the sets. This filter-ID is used as the new search-ID. In the illustrated example, the filter-IDs that have not been searched in the sets are:

Source port = {P7(4000 , 1100, 8100), P8(4000, 1200, 8300), P9(4000, 1300, 8300), P10(4000 ,1400, 8400), P11(4000, 1500, 8300), P12(4000, 2000, 9500)}

Destination port: {{P1(4000, 1000, 8300), P9(4000, 1300, 8300), P2(4100, 1000, 8100), P3(4200, 1000, 8200)}

Source address: {{P1(4000, 1000, 8300), P8(4000, 1200, 8300), P9(4000, 1300, 8300), P11(4000, 1500, 8300)}

The set of largest filter-IDs is {P12(4000, 2000, 9500), P3(4200, 1000, 8200), P11(4000, 1500, 8300)}. The lowest of these, filter-ID = P11(4000, 1500, 8300), is selected as the new search-ID.

The remaining sets are searched for a filter-ID that matches the new search-ID. The source port set of filter-IDs below the previous search-ID is searched

towards the lowest filter-ID for a match to the new search-ID. In this case, the search-ID, P11(4000, 1500, 8300), is not found in all the remaining sets and is not added to the result intersection set.

5 Because the method has not yet exhausted the filter-IDs in any set, the next largest filter-ID that does not exceed the search-ID is selected from the sets. This filter-ID is used as the new search-ID. In the illustrated example, the filter-IDs that have not been
10 searched in the sets are:

Source port = {P7(4000 , 1100, 8100), P8(4000, 1200, 8300), P9(4000, 1300, 8300), P10(4000 ,1400, 8400)}

15 Destination port: {{P1(4000, 1000, 8300), P9(4000, 1300, 8300)}}}

Source address: {{P1(4000, 1000, 8300), P8(4000, 1200, 8300), P9(4000, 1300, 8300)}}

20

The set of largest filter-IDs is {P10(4000 ,1400, 8400), P9(4000, 1300, 8300), P9(4000, 1300, 8300)}. The lowest of these, filter-ID = P9(4000, 1300, 8300), is selected as the new search-ID.

25 The remaining sets are searched for a filter-ID that

matches the search-ID. In the illustrated example, the destination port and the source address sets are searched from the highest towards the lowest filter-ID. The search-ID, P9(4000, 1300, 8300), is found in all the
5 remaining sets and is added to the result intersection set.

Because the method has not yet exhausted the filter-IDs in any set, the next largest filter-ID that does not exceed the search-ID is selected from the sets. This
10 filter-ID is used as the new search-ID. In the illustrated example, the filter-IDs that have not been searched in the sets are:

Source port = {P7(4000 , 1100, 8100), P8(4000, 1200,
15 8300)}

Destination port: {{P1(4000, 1000, 8300)}

Source address: {{P1(4000, 1000, 8300), P8(4000,
1200, 8300)}}

20 The set of largest filter-IDs is {P8(4000, 1200, 8300), {P1(4000, 1000, 8300), P8(4000, 1200, 8300)}}. The lowest of these, filter-ID = P1(4000, 1000, 8300), is selected as the new search-ID.

The remaining sets are searched for a filter-ID that
25 matches the new search-ID. The source port set of

filter-IDs below the previous search-ID is searched towards the lowest filter-ID for a match to the new search-ID. The search-ID, P1(4000, 1000, 8300), is not found in all the remaining sets and is not added to the
5 result intersection set.

The method in the foregoing example has reached the lowest value filter-ID of the source port set, and the method is completed. The result intersection set is {P4(4300, 1000, 8300), P9(4000, 1300, 8300)}.

10 Various features of the system can be implemented in hardware, software, or a combination of hardware and software. For example, some aspects of the system can be implemented in computer programs executing on programmable computers. Each program can be implemented in a high level
15 procedural or object-oriented programming language to communicate with a computer system.

Furthermore, each such computer program can be stored on a storage medium, such as read-only-memory (ROM), readable by a general or special purpose programmable
20 computer, for configuring and operating the computer when the storage medium is read by the computer to perform the functions described above.

Other implementations are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method comprising:

determining respective sets of one or more filter-
identifiers, each set associated with a respective data-
5 packet classifier field; and

deriving a result-set of filter-identifiers based on an
intersection of the filter-identifier sets.

2. The method of claim 1 further comprising:

associating each filter-identifier with a policy-
10 identifier that identifies a policy applicable to data
packets having specified entries in the data-packet
classifier fields.

3. The method of claim 2 wherein associating a filter-
identifier with a policy-identifier comprises:

15 characterizing the specified entries in the data-packet
classifier fields as either a range-based value or an exact
value;

developing a bit mask from said characterizing; and
combining the bit mask with the policy-identifier.

20 4. The method of claim 2 wherein associating a filter-
identifier with a policy-identifier comprises:

assigning a hierarchical order to the data-packet classifier fields; and

combining a value derived from the data-packet classifier fields with a policy-identifier.

5 5. The method of claim 1 wherein deriving the result-set comprises:

selecting a particular filter-identifier from one of the data-packet classifier field sets as a search-identifier;

10 searching the other packet classifier field sets for a match of the search-identifier; and

adding the search-identifier to a result set if the search-identifier is found in all of the other sets.

6. The method of claim 5 further comprising iteratively
15 repeating the selecting, searching and adding with respect to at least one subsequent filter-identifier until a last filter-identifier in any set is reached.

7. The method of claim 6 further comprising:

placing the filter-identifiers in a hierarchical order;
and

performing said searching from a filter-identifier with
5 a highest value towards a filter-identifier with a lowest
value in the hierarchy.

8. The method of claim 7 wherein selecting a particular
filter-identifier comprises:

selecting an initial filter-identifier from each data-
10 packet classifier field set, said initial filter-identifier
having a highest value among the filter-identifiers in that
data-packet classifier field set; and

selecting one of the initial filter-identifiers as the
particular filter-identifier, said one of the initial
15 filter-identifiers having a lowest value among the initial
filter-identifiers.

9. The method of claim 8 wherein iteratively selecting a subsequent filter-identifier comprises:

selecting a filter-identifier from each data-packet classifier field set, said filter-identifier having a
5 highest value among the filter-identifiers in that data-packet classifier field set that is lower in value than the search-identifier value; and
selecting one of the filter-identifiers as the
iterative particular filter-identifier, said one of the
10 iterative filter-identifiers having a lowest value among the iterative filter-identifiers.

10. A method comprising:

entering node values in a tree for each endpoint value associated with at least one policy-identifier;

15 associating a first set of policy-identifiers with a particular one of the node values wherein the first set of policy-identifiers is applied to data-packets having packet-header classifier fields matching the particular one of the node values; and

20 associating a second set of policy-identifiers with the particular node value wherein the second set of policy-identifiers is applied to data-packets having packet-header classifier fields with a value between the particular one of the node values and a next higher node value.

11. The method of claim 10 further comprising deriving the second set of policy-identifiers from an intersection set of the first set of policy-identifiers and a first set of policy-identifiers associated with the next higher node
5 value.

12. An article comprising a machine-readable medium storing machine-executable instructions that when applied to a machine cause the machine to:

determine respective sets of one or more filter-
10 identifiers associated with respective data-packet classifier fields; and

derive a result-set of filter-identifiers that is based on an intersection of the filter-identifier sets.

13. The article of claim 12 further comprising machine-
15 executable instructions for causing the machine to associate each filter-identifier with a policy-identifier that identifies a policy applicable to data packets having specified entries in the data-packet classifier fields.

14. The article of claim 13 wherein associating a filter-identifier with a policy-identifier comprises instructions to:

assign a hierarchical order to the data-packet

5 classifier fields; and

combine a value derived from the data-packet classifier fields with a policy-identifier.

15. The article of claim 13 comprising the machine-executable instructions to cause the machine to:

10 select a particular filter-identifier from one of the data-packet classifier field sets as a search-identifier;

search the other packet classifier field sets for a match of the search-identifier;

add the search-identifier to a result set if the
15 search-identifier is found in all of the other sets; and

iteratively repeat the selection, search and addition with respect to at least one subsequent filter-identifier until a last filter-identifier in any set is reached.

16. The article of claim 15 further comprising machine-executable instructions for causing the machine to:

place the filter-identifiers in a hierarchical order;
and

5 search from a filter-identifier with a highest value towards a filter-identifier with a lowest value in the hierarchy.

17. The article of claim 16 wherein the machine-executable instructions to select a particular filter-identifier

10 further comprise instructions to cause the machine to:

select an initial filter-identifier from each data-packet classifier field set, said initial filter-identifier having a highest value among the filter-identifiers in that data-packet classifier field set; and

15 select one of the initial filter-identifiers as the particular filter-identifier, said one of the initial filter-identifiers having a lowest value among the initial filter-identifiers.

18. The article of claim 17 further comprising the machine-executable instructions for causing the machine to:

select a filter-identifier from each data-packet classifier field set, said filter-identifier having a

5 highest value among the filter-identifiers in that data-packet classifier field set that is lower in value than the search-identifier value; and

select one of the filter-identifiers as the iterative particular filter-identifier, said one of the iterative

10 filter-identifiers having a lowest value among the iterative filter-identifiers.

19. A article comprising a machine-readable medium storing machine-executable instructions that when applied to a machine cause the machine to:

15 enter node values in a tree for each endpoint value associated with at least one policy-identifier;

associate a first set of policy-identifiers with a particular one of the node values wherein the first set of policy-identifiers is applied to data-packets having packet-
20 header classifier fields matching the particular one of the node values; and

associate a second set of policy-identifiers with the particular one of the node values wherein the second set of policy-identifiers is applied to data-packets having packet-header classifier fields with a value between the particular
5 one of the node values and a next higher node value.

20. The article of claim 19 further comprising instructions to derive the second set of policy-identifiers from an intersection set of the first set of policy-identifiers and a first set of policy-identifiers associated with the next
10 higher node value.

21. An apparatus comprising:

a network interface adapter;

first circuitry communicatively coupled to the network interface adapter to determine respective sets of one or
15 more filter-identifiers associated with respective data-packet classifier fields; and

second circuitry communicatively coupled to the network interface adapter to derive a result-set of filter-identifiers that is based on an intersection of the filter-
20 identifier sets.

22. The apparatus of claim 21 further comprising third circuitry communicatively couple to the network interface adapter to associate each filter-identifier with a policy-identifier that identifies a policy applicable to data
5 packets having specified entries in the data-packet classifier fields.

23. The apparatus of claim 22 wherein associating a filter-identifier with a policy-identifier comprises fourth circuitry communicatively coupled to the network interface
10 adapter to:

assign a hierarchical order to the data-packet classifier fields; and

combine a value derived from the data-packet classifier fields with a policy-identifier.

15 24. The apparatus of claim 23 further comprising fifth circuitry communicatively coupled to the network interface adapter to:

select a particular filter-identifier from one of the data-packet classifier field sets as a search-identifier;

20 search the other packet classifier field sets for a match of the search-identifier;

add the search-identifier to a result set if the search-identifier is found in all of the other sets; and

iteratively repeat the selection, search and addition with respect to at least one subsequent filter-identifier until a last filter-identifier in any set is reached.

25. The apparatus of claim 24 further comprising sixth
5 circuitry communicatively coupled to the network interface adapter to:

place the filter-identifiers in a hierarchical order;
and

search from a filter-identifier with a highest value
10 towards a filter-identifier with a lowest value in the hierarchy.

26. The apparatus of claim 25 further comprising seventh circuitry communicatively coupled to the network interface adapter to:

15 select an initial filter-identifier from each data-packet classifier field set, said initial filter-identifier having a highest value among the filter-identifiers in that data-packet classifier field set; and

select one of the initial filter-identifiers as the
20 particular filter-identifier, said one of the initial filter-identifiers having a lowest value among the initial filter-identifiers.

27. The apparatus of claim 26 further comprising eighth circuitry communicatively coupled to the network interface adapter to:

select a filter-identifier from each data-packet classifier field set, said filter-identifier having a highest value among the filter-identifiers in that data-packet classifier field set that is lower in value than the search-identifier value; and

select one of the filter-identifiers as the iterative particular filter-identifier, said one of the iterative filter-identifiers having a lowest value among the iterative filter-identifiers.

28. A system comprising:

a network; and
a network device communicatively coupled to the network, the device including a processor and instructions executable by the processor to cause the processor to:

determine respective sets of one or more filter-identifiers associated with respective data-packet classifier fields; and

derive a result-set of filter-identifiers that is based on an intersection of the filter-identifier sets.

29. The system of claim 28 further comprising instructions to cause the processor to associate each filter-identifier with a policy-identifier that identifies a policy applicable to data packets having specified entries in the data-packet classifier fields.

30. The system of claim 29 further comprising instructions to cause the processor to:

select a particular filter-identifier from one of the data-packet classifier field sets as a search-identifier;

10 search the other packet classifier field sets for a match of the search-identifier;

add the search-identifier to a result set if the search-identifier is found in all of the other sets; and

iteratively repeat the selection, search and addition with respect to at least one subsequent filter-identifier until a last filter-identifier in any set is reached.