Access Control Lists (ACLs)
FEATURE OVERVIEW AND CONFIGURATION GUIDE

Introduction

This guide describes Access Control Lists (ACLs), and general ACL configuration information. For detailed command information and examples for ACL commands, see the following chapters in your product’s Command Reference:

- IPv4 Hardware Access Control List (ACL) Commands
- IPv6 Hardware Access Control List (ACL) Commands
- IPv4 Software Access Control List (ACL) Commands
- IPv6 Software Access Control List (ACL) Commands

The Command Reference is available on our website at alliedtelesis.com.

Hardware ACLs are applied directly to interfaces, or are used for QoS classifications. Software ACLs are applied to Routing and Multicasting.

Products and software version that apply to this guide

This Guide applies to AlliedWare Plus™ products that support ACLs, running version 5.4.4 or later. However, support and implementation of ACLs varies between products. To see whether a product supports a particular feature or command, see the following documents:

- The product’s Datasheet
- The AlliedWare Plus Datasheet
- The product’s Command Reference

These documents are available from the above links on our website at alliedtelesis.com.

Feature support may change in later software versions. For the latest information, see the above documents.
Content

Introduction.............................................................................................................................................................................1
Products and software version that apply to this guide.................................................................1
Overview......................................................................................................................................................................... 3
ACL rules.........................................................................................................................................................................3
Hardware and Software ACL Types.........................................................................................................................4
Numbered ACLs (for hardware and software ACLs)...........................................................................4
Defining hardware MAC ACLs........................................................................................................................... 6
Defining hardware IP ACLs.............................................................................................................................7
Defining Named Hardware ACLs.................................................................................................................... 8
Defining hardware IPv6 ACLs........................................................................................................................... 10
Actions for hardware ACLs............................................................................................................................10
Attaching hardware ACLs to interfaces.................................................................................................11
Hardware ACLs and QoS Classifications............................................................................................................11
QoS ACLs...................................................................................................................................................................11
Attaching hardware ACLs Using QoS............................................................................................................12
Expanding ACL match criteria with QoS.................................................................................................13
Using QoS match commands with TCP flags.......................................................................................14
Profile Limitations on SBx908 and x900 Series Switches........................................................................16
Hardware filter example......................................................................................................................................17
Maximum number of hardware ACLs...........................................................................................................18
Viewing the number of hardware ACLs and bytes used........................................................................19
Filter Limitations for SBx8100 Series Switches......................................................................................20
ACL Filter Sequence Numbers........................................................................................................................21
ACL filter sequence number behavior...........................................................................................................21
ACL filter sequence number applicability.................................................................................................22
ACL filter sequence number types................................................................................................................22
ACL Filter Sequence Configuration..................................................................................................................23
Creating ACLs in Global Configuration mode........................................................................................25
Display the ACL configuration details........................................................................................................27
ACL source and destination addresses.......................................................................................................28
ACL reverse masking.........................................................................................................................................28
Overview

An Access Control List is one filter, or a sequence of filters, that are applied to an interface to either block or pass (or when using QoS, apply priority to) packets that match the filter definitions. ACLs are used to restrict network access by hosts and devices and to control network traffic.

An ACL contains an ordered list of filters. Each filter specifies either permit or deny and a set of conditions the packet must satisfy in order to match the filter. The meaning of permit or deny entries depends on the context in which the ACL is used - either on an inbound or an outbound interface.

When a packet is received on an interface, the switch compares fields in the packet against filters in the ACL to check whether the packet has permission to be forwarded, based on the filter properties. The comparison process stops as soon as the first match is found, and then the action of the ACL is applied. If no entries match, then, for the case of AlliedWare Plus hardware ACLs, the ACL ends in an implicit ‘permit all else’ clause. So, the unmatched packets are permitted.

Because filters in an ACL are applied sequentially and their action stops at the first match, it is very important that you apply the filters in the correct order. For example you might want to pass all traffic from VLAN 4 except for that arriving from two selected addresses A and B. Setting up a filter that first passes all traffic from VLAN 4 then denies traffic from addresses A and B will not filter out traffic from A and B if they are members VLAN 4. To ensure that the traffic from A and B is always blocked you should first apply the filter to block traffic from A and B, then apply the filter to allow all traffic from VLAN 4. You can assign sequence numbers to filters. See "ACL Filter Sequence Numbers" on page 21 for more information.

ACL rules

- The source or destination address or the protocol of each packet being filtered are tested against the filters in the ACL, one condition at a time (for a permit or a deny filter).
- If a packet does not match a filter then the packet is checked against the next filter in the ACL.
- If a packet and a filter match, the subsequent filters in the ACL are not checked and the packet is permitted or denied as specified in the matched filter.
- The first filter that the packet matches determines whether the packet is permitted or denied. After the first match, no subsequent filters are considered.
- If the ACL denies the address or protocol then the software discards the packet.
- For hardware ACLs, if no filters match then the packet is forwarded.
- Checking stops after the first match, so the order of the filters in the ACL is critical. The same permit or deny filter specified in a different order could result in a packet being passed in one situation and denied in another situation.
- Multiple ACLs per interface, per protocol (i.e. IPv4 and IPv6), per direction are allowed.
- For inbound ACLs, a permit filter continues to process the packet after receiving it on an inbound interface, and a deny filter discards the packet.
Hardware and Software ACL Types

Access Control Lists used in AlliedWare Plus are separated into two different types, software ACLs and hardware ACLs. You can define both types as either named or numbered.

**Note:** The filtering principles applied to software ACLs (those in the range 1 to 2699) are different to those applied to hardware ACLs (those in the range 3000 to 4699).
- software ACLs will *deny* access unless *explicitly permitted* by an ACL action.
- hardware ACLs will *permit* access unless *explicitly denied* by an ACL action.

Numbered ACLs (for hardware and software ACLs)

Numbered ACLs are assigned an ACL number within the range 1 to 4699. ACL numbers are grouped into ranges, where each range denotes a specific functionality. The following table shows the number ranges and functionality that your switch supports.

**Table 1: ACL numeric ranges and functionality**

<table>
<thead>
<tr>
<th>ACL NUMBER RANGE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 99</td>
<td>IP standard ACL</td>
</tr>
<tr>
<td>100 to 199</td>
<td>IP extended ACL</td>
</tr>
<tr>
<td>1300 to 1999</td>
<td>IP standard expanded ACL</td>
</tr>
<tr>
<td>2000 to 2699</td>
<td>IP extended expanded ACL</td>
</tr>
<tr>
<td>3000 to 3699</td>
<td>Hardware IP ACL</td>
</tr>
<tr>
<td>4000 to 4699</td>
<td>Hardware MAC ACL</td>
</tr>
</tbody>
</table>

Software ACLs that use either the ranges 1-99, 100-199, 1300-1999, 2000-2699, or are named ACLs (that use the standard or extended keyword followed by a text string), are used in features such as SNMP, IGMP and OSPF.

Hardware ACLs

These ACL types are applied directly to an interface, or are used for QoS classifications.

Hardware ACLs use the following ranges:

- 3000-3699 for hardware IP ACLs
- 4000-4699 for hardware MAC ACLs
- named hardware IPv4 ACLs
- named hardware IPv6 ACLs
Software ACLs

These ACLs types can be either named ACLs, using the standard or extended keyword followed by a text string.

Software ACLs use the following ranges:

- 1-99 (IP standard ACL range)
- 100-199 (IP extended ACL range)
- 1300-1999 (IP standard expanded ACL range)
- 2000-2699 (IP extended expanded ACL range)
- named standard IPv4 ACLs
- named extended IPv4 ACLs
- named standard IPv6 ACLs
- named extended IPv6 ACLs

In AlliedWare Plus, software ACLs are not used directly for filtering of packets that are being forwarded.

Rather, the software ACLs are used more for purposes like defining:

- ranges of addresses which to which protocol parameters are applied
- ranges of address which are excluded from being operated on by protocols
- routes to be included/excluded in the operation of routing protocols

The types of features that make use of software ACLs for these purposes are SNMP, PIM, IGMP, OSPF, BGP.

Examples of the use of software ACLs are:

- Specifying a set of RIP routes to which a particular Administrative Distance should be applied:
  ```
  awplus(config)# distance <1-255> ip/mask <access-list>
  ```

- To filter which routes from the OSPF route table should be imported into the main IP route table:
  ```
  awplus(config)# distribute-list <access-list> in
  ```

- To define the addresses of management stations that can access a given SNMP community:
  ```
  awplus(config)# access-list 66 permit 192.168.11.5
  awplus(config)# snmp-server community example1rw rw 66
  ```

- To specify the range of multicast groups for which a router is offering PIM RP candidacy:
  ```
  awplus(config)# ip pim rp-candidate <interface>[priority <priority>][interval <interval>][grouplist <grouplist>]
  ```
Defining hardware MAC ACLs

These are used to filter traffic based on specific source or destination MAC addresses contained within the data frames. They can be applied to ports in the form of access groups.

A MAC access list requires the following components:

- an ACL number in the range 4000-4699.
- an action, permit, or deny. See "Actions for hardware ACLs" on page 10
- a source MAC address. You can use the format, HHHH.HHHH.HHHH to filter on a specific MAC address (where H is a hexadecimal number), or you can filter on any source MAC address by entering the word “any”.
- a source MAC mask. This mask determines which portion of the source MAC address header will be compared with that found in the incoming packets. The mask is configured in the format <HHHH.HHHH.HHHH> where each H is a hexadecimal number. In practice each hex number will normally be either 0 (to represent a match) or F (to represent a don’t care condition). A mask is not required if the source address is specified as “any”.
- a destination MAC address. You can use the format, HHHH.HHHH.HHHH to filter on a specific MAC address (where H is a hexadecimal number), or you can filter on any destination MAC address by entering the word “any”.
- a destination MAC mask. This mask determines which portion of the destination MAC address header will be compared with that found in the incoming packets. The mask is configured in the format <HHHH.HHHH.HHHH> where each H is a hexadecimal number. In practice each hex number will normally be either 0 (to represent a match) or F (to represent a don’t care condition). A mask is not required if the source address is specified as “any”.

Example

To permit packets coming from a specific MAC address of 0030.841A.1234 and with any destination address:

awplus# configure terminal
awplus(config)# access-list 4000 permit 0030.841A.1234 0000.0000.0000 any
Defining hardware IP ACLs

Hardware IP ACLs are used to filter traffic based on specific source or destination IP addresses and/or Layer 4 parameters contained within the data frames. They can be applied to ports in the form of access groups.

An IP access list requires the following components:

- an ACL number in the range 3000-3699
- an action, see "Actions for hardware ACLs" on page 10
- a packet type:
  - IP: This matches any type of IP packet. A source and destination address must be specified, although they can be “any”. The source address matches packets coming from specified networking devices or hosts. The destination address matches packets going to specified networking devices or hosts.
  - ICMP: This matches ICMP packets. A source and destination address must be specified, although they can be “any”. An ICMP type can optionally be specified after the destination address.
  - TCP: This matches TCP packets. A source and destination address must be specified, although they can be “any”. After the source address, a source port can optionally be specified and after the destination address a destination port can optionally be specified. The port matching can be done using `eq` (equal to), `gt` (greater than), `lt` (less than), `ne` (not equal to), or `range` (for a range of ports, which requires a start port and an end port).
  - UDP: This matches UDP packets and has the same options as TCP.
  - proto: This allows any IP protocol type to be specified (e.g. 89 for OSPF). A source and destination address must be also specified, although they can be “any”.

For example:

- To match (and permit) any type of IP packet containing a destination address of 192.168.1.1
  ```
  awplus(config)# access-list 3000 permit ip any 192.168.1.1/32
  ```

- To match (and permit) an ICMP packet with a source address of 192.168.x.x and an ICMP code of 4
  ```
  awplus(config)# access-list 3001 permit icmp 192.168.0.0/16 any icmp-type 4
  ```

- To match a TCP packet with a source address of 192.168.x.x, source port of 80 and a destination port from 100 to 150:
  ```
  awplus(config)# access-list 3002 permit tcp 192.168.0.0/16 eq 80 any range 100 150
  ```

- To match a UDP packet with a source address of 192.168.x.x, a destination address of 192.168.1.x, and a destination port greater than 1024:
  ```
  awplus(config)# access-list 3003 permit udp 192.168.0.0/16 192.168.1.0/24 gt 1024
  ```
To match (and permit) a UDP packet with a source address of 192.168.30.2/32 and a destination port of 5062:

```
awplus(config)# access-list 3002 permit udp any 192.168.30.2/32 eq 5062
```

To match to any OSPF packet

```
awplus(config)# access-list 3004 permit proto 89 any any
```

**Note:** An IP address mask can be specified using either of the following notations:

- A.B.C.D/M: This is the most common: e.g. 192.168.1.0/24
- A.B.C.D A.B.C.D: 192.168.1.1 0.0.0.0 is the same as 192.168.1.1/32 and 192.168.1.1 255.255.255.255 is the same as “any”

ACLs use reverse masking, also known as wildcard masking, to indicate to the switch whether to check or ignore corresponding IP address bits when comparing the address bits in an ACL filter to a packet being submitted to the ACL.

Reverse masking for IP address bits specifies how the switch treats the corresponding address bits. A reverse mask is also called an inverted mask because a 1 and 0 mean the opposite of what they mean in a subnet or a network mask.

- A reverse mask bit 0 means check the corresponding bit value.
- A reverse mask bit 1 means ignore the corresponding bit value.

- host A.B.C.D: This is the same as A.B.C.D/32

### Defining Named Hardware ACLs

A Named Sequential Hardware ACL consists of a series of filter entries. The only limit on the number of filter entries that you can add to the ACL is on the number of entries that can fit into the hardware table - the software does not put any other lower limit on the number of entries.

Entries in the ACL can be from four different types:

1. **IP protocol filter entry**

   This can match on any combination of the fields:
   - IP protocol number; for example 1 for ICMP, 2 for IGMP, 50 for ESP, 89 for OSPF, etc. Or, you can simply specify "IP", to match any IP protocol
   - Source IP address—an individual IP address or a subnet
   - Dest IP address—an individual IP address or a subnet
   - Source MAC address—an individual MAC address or a range
   - Dest MAC address—an individual MAC address or a range
   - VLAN ID
2. **MAC filter entry**

   This can match on any combination of the fields:
   - Source MAC address—an individual MAC address or a range
   - Dest MAC address—an individual MAC address or a range
   - VLAN ID
   - Inner VLAN ID

3. **ICMP protocol filter entry**

   This can match on any combination of the fields:
   - Source IP address—an individual IP address or a subnet
   - Dest IP address—an individual IP address or a subnet
   - ICMP type
   - VLAN ID

4. **TCP/UDP protocol filter entry**

   - Source IP address—an individual IP address or a subnet
   - Dest IP address—an individual IP address or a subnet
   - Source TCP/UDP port—either single port number or a range
   - Dest TCP/UDP port—either single port number or a range
   - VLAN ID

You can find the exact syntax of the commands to create these entries in the x900/x908 software reference manual.

There is no restriction on the combinations of filter entry types that can exist together in the same

**To define a Named IPv4 Hardware ACL**

**Step 1: Create the ACL**

```
awplus(config)# access-list hardware hw_name
```

This will put you into Hardware ACL Configuration mode, with the prompt:

```
awplus(config-ip-hw-acl)#
```

**Step 2: Create the individual filter entries within the ACL**

```
awplus(config-ip-hw-acl)# permit ip 192.168.1.0 0.0.0.255
192.168.2.0 0.0.0.255
awplus(config-ip-hw-acl)# deny ip 192.168.1.0 0.0.0.255 any
```

and so on...
Defining hardware IPv6 ACLs

There are no numbered IPv6 hardware ACLs. The only form of IPv6 hardware ACLs available in AlliedWare Plus are Named IPv6 hardware ACLs.

To create an IPv6 hardware ACL

**Step 1: Create the ACL with the command**

```
awplus(config)# ipv6 access-list ipv6_hw_name
```

This puts you into IPv6 hardware configuration mode, with the prompt:

```
awplus(config-ipv6-hw-acl)#
```

**Step 2: Define the filters that comprise the content of the ACL**

```
awplus(config-ipv6-hw-acl)# permit ip 2001:db8::/64 2001:db9::/64
awplus(config-ipv6-hw-acl)# deny ip 2001:db8::/64 any
```

Actions for hardware ACLs

Depending on your switch family, the following actions are available for hardware ACLs:

**Table 2: Hardware ACL parameter actions**

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>deny</td>
<td>Discard the packet</td>
</tr>
<tr>
<td>permit</td>
<td>Allow the packet</td>
</tr>
<tr>
<td>copy-to-cpu</td>
<td>Send a copy of the packet to the CPU and forward it as well. This is the same as copy, forward in AW hardware filters.</td>
</tr>
<tr>
<td>send-to-cpu</td>
<td>Send the packet to the CPU and do not forward it. This is the same as copy, discard in AlliedWare hardware filters.</td>
</tr>
<tr>
<td>send-to-mirror</td>
<td>Send the packet to the mirror port so packets are not switched.</td>
</tr>
<tr>
<td>send-to-cpu</td>
<td>Send the packet to the CPU and do not forward it. Note that specifying this action could result in EPSR healthcheck messages and other control packets being dropped.</td>
</tr>
<tr>
<td>copy-to-mirror</td>
<td>Send a copy of the packet to the mirror port and forward it as well.</td>
</tr>
</tbody>
</table>
Attaching hardware ACLs to interfaces

A hardware ACL is attached directly to a switchport using the `access-group` command. For example, to permit traffic from 192.168.1.x, but discard from 192.168.x.x:

```
awplus# configure terminal
awplus(config)# access-list 3000 permit ip 192.168.1.0/24 any
awplus(config)# access-list 3001 deny ip 192.168.0.0/24 any
awplus(config)# interface port1.0.1
awplus(config-if)# access-group 3000
awplus(config-if)# access-group 3001
```

Similarly, a named hardware ACL can be applied to an interface with the commands:

```
awplus(config)# interface port1.0.1
awplus(config-if)# access-group <ACL-name>
```

Hardware ACLs and QoS Classifications

Interface ACLs and QoS policies can both be attached to the same port. Where this is done, packets received on the port will be matched against the ACLs first.

The interface ACLs and QoS classifications are implemented by taking the first matching filter and applying the action defined for that filter. All subsequent matches in the table are then ignored. Thus, because ACLs are also matched first, if the matching ACL has a permit action, the packet is forwarded due to that rule's action and any subsequent QoS rules are bypassed.

You can also apply permit rules using QoS.

For example, you might want to permit a source IP address of 192.168.1.x, but block everything else on 192.168.x.x.

In this case you could create both the permit and deny rules using QoS.

For more information on these applications see "Actions for hardware ACLs" on page 10

QoS ACLs

When using ACLs through QoS, the same classification and action abilities are available, but QoS has some additional fields that it can match on, see "Expanding ACL match criteria with QoS" on page 13, and also provides the ability to perform metering, marking, and remarking on packets that match the filter definitions.

The action used by a QoS class-map is determined by the ACL that is attached to it. If no ACL is attached, it uses the permit action. If an ACL is not required by the class-map (for
example, only matching on the VLAN) and a deny action is required, a MAC ACL should be added with any for source address and any for destination address.

The following example creates a class-map with will deny all traffic on VLAN 2:

```
awplus(config)# access-list 4000 deny any any
awplus(config)# class-map cmap1
awplus(config-cmap)# match access-group 4000
awplus(config-cmap)# match vlan 2
```

The default class-map matches to all traffic and so cannot have any match or ACL commands applied to it. The action for this class-map is set via the default-action command and is permit by default. It can be changed to deny by using the following commands:

```
awplus(config)# policy-map pmap1
awplus(config-pmap)# default-action deny
```

For more information on applying QoS filtering, see the QoS Feature Overview and Configuration Guide.

### Attaching hardware ACLs Using QoS

The same functionality can be achieved using QoS, by attaching the ACL to a class-map, attaching the class-map to a policy-map and attaching the policy-map to a port:

**Step 1:** Enable QoS on the switch

```
awplus(config)# mls qos enable
```

**Step 2:** Create access lists

Create ACL **3000** to permit all packets from the 192.168.1 subnet:

```
awplus(config)# access-list 3000 permit ip 192.168.1.0/24 any
```

Create ACL **3001** to deny all packets from the 192.168.0 subnet:

```
awplus(config)# access-list 3001 deny ip 192.168.0.0/24 any
```

**Step 3:** Attach access-groups to class-maps

Attach ACL **3000** to the class-map `cmap1`:

```
awplus(config)# class-map cmap1
awplus(config-cmap)# match access-group 3000
awplus(config-cmap)# exit
```

Attach ACL **3001** to another class-map (`cmap2`)

```
awplus(config)# class-map cmap2
awplus(config-cmap)# match access-group 3001
awplus(config-cmap)# exit
```
Step 4: Attach class-maps to policy-maps

Attach the class-map cmap1 to policy-map pmap1:
awplus(config)# policy-map pmap1
awplus(config-pmap)# class cmap1
awplus(config-pmap-c)# exit

Add the class-map cmap2 to the policy-map pmap1:
awplus(config-pmap)# class cmap2
awplus(config-pmap-c)# exit

Return to Global Configuration mode
awplus(config-pmap)# exit

Step 5: Add policy-maps to ports

Add policy-map pmap1 to port1.0.1
awplus(config)# interface port1.0.1
awplus(config-if)# service-policy input pmap1

Only one ACL can be attached to a class-map, but multiple class-maps can be attached to a policy-map. Interface ACLs can be attached to the same port as a QoS policy, with the interface ACLs being matched first as described at the beginning of the Classification section.

Expanding ACL match criteria with QoS

Another reason for using QoS rather than interface ACLs is that QoS provides a lot more fields on which to match. These are accessed through the match commands in config-cmap mode.

Config-cmap mode describes the fields that can be matched on. Only one of each type can be matched, with the exception of tcp-flags (see below for classification). If multiple matches are specified, they are ANDed together. The following example shows how you can match a packet on VLAN 2, that has a source IP address of 192.168.x.x and a DSCP of 12:

1. Create ACL 3000 to permit all packets from the 192.168 subnet:
awplus# configure terminal
awplus(config)# access-list 3000 permit ip 192.168.0.0/16 any

2. Apply ACL 3000 to the class-map cmap1, add the matching criteria of VLAN 2 and DSCP 12:
awplus(config)# class-map cmap1
awplus(config-cmap)# match access-group 3000
awplus(config-cmap)# match vlan 2
awplus(config-cmap)# match dscp 12
awplus(config-cmap)# exit
Using QoS match commands with TCP flags

Usually, if multiple matches of the same type are specified, the matching process will apply to the last match that you specified. For TCP flags however, the arguments are ANDed together. For example, the following series of commands will match on a packet that has `ack`, `syn`, and `fin` set:

```
awplus# configure terminal
awplus(config)# class-map cmap1
awplus(config-cmap)# match tcp-flags ack
awplus(config-cmap)# match tcp-flags syn
awplus(config-cmap)# match tcp-flags fin
awplus(config-cmap)# exit
```

The following commands will achieve the same result:

```
awplus# configure terminal
awplus(config)# class-map cmap1
awplus(config-cmap)# match tcp-flags ack syn fin
awplus(config-cmap)# exit
```

Note that the matching is looking to see whether “any” of the specified flags are set. There is no checking for whether any of these flags are unset. Therefore the following commands will match on a packet in any of the following combinations of syn and ack status flags as shown in the following table:

```
SYN  | ACK  | MATCH ON PACKET
Set  | Set  | Yes
Set  | Unset| Yes
Unset| Set  | No
Unset| Unset| No
```

If you want to drop packets with syn only, but not with ack and syn, the following two class-maps can be used (note that ACL 4000 is used to apply a drop action as described in “Actions for hardware ACLs” on page 10):
**Step 1: Create access lists**

Create ACL **4000** to deny all packets with any source or destination address

```
awplus# configure terminal
awplus(config)# access-list 4000 deny any any
```

**Step 2: Create class-maps**

Create the class-map **cmap1** and configure it to match on the TCP flags, **ack** and **syn**:

```
awplus(config)# class-map cmap1
awplus(config-cmap)# match tcp-flags ack syn
awplus(config-cmap)# exit
```

Create the class-map **cmap2** and configure it to match on the TCP flag, **syn**

```
awplus(config)# class-map cmap2
awplus(config-cmap)# match tcp-flags syn
awplus(config-cmap)# exit
```

**Step 3: Apply access-groups to class-maps**

Apply ACL **4000** to this class-map (i.e. to **cmap2**):

```
awplus(config-cmap)# match access-group 4000
awplus(config-cmap)# exit
```

**Step 4: Create policy-maps**

Create the policy-map **pmap1** and associate it with **cmap1**:

```
awplus(config)# policy-map pmap1
awplus(config-pmap)# class cmap1
awplus(config-pmap-c)# exit
```

**Step 5: Associate class-maps with policy-maps**

Associate **cmap2** with this policy-map (**pmap1**)

```
awplus(config-pmap)# class cmap2
awplus(config-pmap-c)# exit
```
Profile Limitations on SBx908 and x900 Series Switches

On SBx908 and x900 Series switches, a profile is a mask that comprises 16 bytes. Each filter item that is added to the ACL set will consume a portion of the 16 bytes. Note that Hardware ACLs and QoS filters both share this single mask. However each time a mask component is defined within the mask, it can by used in many ACLs - so it is the number of different components that is important.

The following table will help you manage your ACL mask.

Table 3: ACL mask components

<table>
<thead>
<tr>
<th>PROTOCOL COMPONENT</th>
<th>BYTES USED IN THE MASK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol Type</td>
<td>2</td>
</tr>
<tr>
<td>Ethernet format</td>
<td>2</td>
</tr>
<tr>
<td>IP protocol type (e.g. TCP, UDP)</td>
<td>1</td>
</tr>
<tr>
<td>source IPv4 address(^1)</td>
<td>0</td>
</tr>
<tr>
<td>destination IPv4 address(^1)</td>
<td>0</td>
</tr>
<tr>
<td>source IPv6 address(^1)</td>
<td>0</td>
</tr>
<tr>
<td>destination IPv6 address(^1)</td>
<td>10</td>
</tr>
<tr>
<td>Ipv6 Next Header</td>
<td>1</td>
</tr>
<tr>
<td>TCP port number</td>
<td>2</td>
</tr>
<tr>
<td>UDP port number</td>
<td>2</td>
</tr>
<tr>
<td>MAC source address</td>
<td>6</td>
</tr>
<tr>
<td>MAC destination address</td>
<td>6</td>
</tr>
<tr>
<td>Inner Vlan ID (includes two bytes TPID)</td>
<td>4</td>
</tr>
<tr>
<td>Inner CoS (includes two bytes TPID)</td>
<td>3</td>
</tr>
<tr>
<td>Tag Protocol Identifier (TPID)</td>
<td>2</td>
</tr>
<tr>
<td>Inner Tag Protocol Identifier (TPID)</td>
<td>2</td>
</tr>
<tr>
<td>IP precedence value</td>
<td>1</td>
</tr>
</tbody>
</table>

---

Most classification fields use some of the 16 'profile bytes' that are available to classify incoming packets. For example, an ACL with both source and destination MAC address fields set will consume 12 of the 16 bytes. However IPv6 source address, and IPv4 source and destination address, will not consume any bytes, allowing greater freedom in ACL configuration options.
Hardware filter example

If you make a hardware filter that matches on destination IP address and source TCP port, this adds 3 bytes to the mask:

- 1 byte for the IP protocol field (to indicate TCP)
- 2 bytes for the source TCP port number.

If you now create the following (additional) hardware filters:

- A hardware filter that matches on source MAC address: this adds 6 more bytes to the mask.
- A QoS class map that matches on destination IP address and DSCP (1 byte): this adds 1 more byte to the mask, for the DSCP.
- A hardware filter that matches on source IP address and source TCP port: this does not change the mask, because the switch already matches on source TCP port, and source IP address does not use any bytes.
- A hardware filter that matches on source UDP port: this does not add any length to the mask, because it shares the same 2 bytes as the source TCP port. However, if you next make a hardware filter that matches on destination TCP or UDP port, that uses another 2 bytes.

Are there now enough bytes for your set of filters?

The mask has a maximum size of 16 bytes. When it reaches the 16-byte limit, no more classifiers can be used that would cause the mask to increase in size. The switch can still accept classifiers that use fields that have already been included in the mask.

There is no particular number of hardware filters or QoS flow groups that will cause the mask to reach its 16-byte limit - it could happen after a few filters, or you might be able to create hundreds of filters without the mask reaching its limit.

So to determine whether you will have enough filter length, look at the fields you want to filter, determine the number of bytes for each field, and sum up the total number of bytes. If that number is less than 16, there is enough filter length. Don’t forget to count TCP and UDP source port as a single field, and likewise to count TCP and UDP destination port as a single field.
Maximum number of hardware ACLs

The maximum number of individual hardware ACLs depends on the hardware type of the port you apply the ACLs to, and the setting of the `platform hwfilter-size` command.

On SBx908 and x900 Series switches, two hardware types exist: Hardware Version 1 and Hardware Version 2. The following table shows the hardware modules and the version to which each belongs.

**Table 4: Hardware types and modules**

<table>
<thead>
<tr>
<th>HARDWARE VERSION 1</th>
<th>HARDWARE VERSION 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseboard</td>
<td>AT-XEM-12Sv2</td>
</tr>
<tr>
<td>AT-XEM-12T</td>
<td>AT-XEM-12Tv2</td>
</tr>
<tr>
<td>AT-XEM-12XS</td>
<td>AT-XEM-2XP</td>
</tr>
<tr>
<td>AT-XEM-1XP</td>
<td>AT-XEM-2XS</td>
</tr>
<tr>
<td></td>
<td>AT-XEM-2XT</td>
</tr>
<tr>
<td></td>
<td>AT-XEM-24T</td>
</tr>
</tbody>
</table>

The following table shows the maximum number of filters for each hardware type and `platform hwfilter-size` command setting.

**Table 5: Hardware types and filters**

<table>
<thead>
<tr>
<th>HARDWARE TYPE</th>
<th>SETTING OF PLATFORM HWFILTER-SIZE</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Type 1</td>
<td>ipv4-full-ipv6 (Default. Can filter on MAC addresses and IPv4 and IPv6 settings)</td>
<td>1024</td>
</tr>
<tr>
<td>Hardware Type 1</td>
<td>basic (Can only filter on MAC addresses and IPv4 settings)</td>
<td>2048</td>
</tr>
<tr>
<td>Hardware Type 2</td>
<td>ipv4-full-ipv6</td>
<td>4096</td>
</tr>
<tr>
<td>Hardware Type 2</td>
<td>basic</td>
<td>8192</td>
</tr>
</tbody>
</table>
Viewing the number of hardware ACLs and bytes used

The number of hardware ACLs and bytes used are displayed in the output from the `show platform classifier statistics utilization brief` command, as shown in Figure 1.

**Figure 1: Example output from the `show platform classifier statistics utilization brief` command**

<table>
<thead>
<tr>
<th>Instance 0</th>
<th>[port1.0.1-port1.0.12]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of PCE Entries:</td>
<td></td>
</tr>
<tr>
<td>Note: Total available rules depends on routing ratio setting</td>
<td></td>
</tr>
<tr>
<td>Used / Total</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>MC V6</td>
<td>128 (‘ipv6 multicast-routing’ uses this)</td>
</tr>
<tr>
<td>System</td>
<td>0</td>
</tr>
<tr>
<td>DHCP Snooping</td>
<td>0</td>
</tr>
<tr>
<td>Web Auth</td>
<td>0</td>
</tr>
<tr>
<td>Loop Detection</td>
<td>0</td>
</tr>
<tr>
<td>EPSR</td>
<td>0</td>
</tr>
<tr>
<td>SNAP</td>
<td>0</td>
</tr>
<tr>
<td>Global ACL</td>
<td>0</td>
</tr>
<tr>
<td>ACL</td>
<td>0</td>
</tr>
<tr>
<td>QoS</td>
<td>0</td>
</tr>
<tr>
<td>RA Guard</td>
<td>0</td>
</tr>
<tr>
<td>MLD Snooping</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>128 / 1024 (12.50%)</td>
</tr>
<tr>
<td>UDB Usage:</td>
<td></td>
</tr>
<tr>
<td>Legend of Offset Type) 1:Ether 2:IP 3:TCP/UDP</td>
<td></td>
</tr>
<tr>
<td>UDB Set</td>
<td>Offset Type</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>IPv4_TCP</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>IPv4_UDP</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>MPLS</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>IPv4_Frag</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>IPv4</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>Ethernet</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>IPv6</td>
<td>0000000000000000</td>
</tr>
</tbody>
</table>
Filter Limitations for SBx8100 Series Switches

On SBx8100 Series switches, typically, for each ACL or class-map, one filter goes into hardware. The exceptions are when:

- TCP and UDP port ranges are specified. For example, with the `lt`, `gt`, `ne`, and `range` parameters of the `access-list` command. In which case, there might be multiple filters created, in order to cover the specified range of port numbers.

- a rule is neither IPv6 nor non-IPv6 specific, in which case two filters are added to hardware, one for IPv6 and another for non-IPv6. An example of this is the hardware MAC numbered option in the `access-list` command, which only matches on MAC address.

A filter is comprised of standard and optional fields. The standard fields, such as source and destination IP and MAC addresses, are “permanent” in that they are always generated from the packet, whereas the optional fields must share a pool of six defined offsets within the packet. The offset types are displayed in the output from the `show platform classifier statistics utilization brief` command, as shown in Figure 2. The optional fields are shown in Table 6 and these optional fields share a limited pool of 6 bytes. Note that all the configured filters share the same offset bytes. However, each offset type can be used in many ACLs.

**Figure 2: Example output from the** `show platform classifier statistics utilization brief` **command**

```
Card 2:

[Instance 1]
[port1.2.1-port1.2.24]          Used / Total
--------------------------------
MLD Snooping          0
DHCP Snooping         0
Web Auth              0
Loop Detection        0
EPSR                  0
IPv6 Global ACL      0
IPv6 ACL              0
Global ACL           0
ACL                   0
QoS                   0
RA Guard              0
Total                  0 / 1536 (0.00%)

UDB Usage:
Legend of Offset Type) 1:Ether 2:IP 3:TCP/UDP
UDB Set  Offset Type  Used / Total
------------- 0------8------15  ------------
Non IPv6      100000          0 /  6
IPv6 L2       220000          0 /  6
```
Different fields in the filter are set and active depending on the settings of the class-map or hardware ACL it represents. Each filter is matched against the fields, standard and optional, taken from each ingressing packet. The first matching filter determines the action taken on the packet.

**Table 6: The user defined optional fields of a filter**

<table>
<thead>
<tr>
<th>PROTOCOL COMPONENT</th>
<th>BYTES USED IN THE FILTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP precedence value</td>
<td>1</td>
</tr>
<tr>
<td>Tag Protocol Identifier (TPID)</td>
<td>2</td>
</tr>
<tr>
<td>ICMP packet type</td>
<td>1</td>
</tr>
<tr>
<td>ICMPv6 packet type</td>
<td>1</td>
</tr>
<tr>
<td>inner VLAN ID</td>
<td>2</td>
</tr>
<tr>
<td>inner CoS</td>
<td>1</td>
</tr>
<tr>
<td>inner Tag Protocol Identifier (TPID)</td>
<td>2</td>
</tr>
<tr>
<td>SNAP tagged and untagged packets</td>
<td>2 or 3, depending on line card</td>
</tr>
</tbody>
</table>

For SBx81GP24 and SBx81GT24 line cards, 2 bytes are used in the filter. For SBx81GS24a and SBx81XS6 line cards, 3 bytes are used in the filter.

**Note:** For SBx81GP24 and SBx81GT24 line cards, a maximum of 1536 filters can be created per line card. For SBx81GS24a, and SBx81XS6 line cards, a maximum of 7168 filters can be created per line card.

### ACL Filter Sequence Numbers

To help you manage ACLs you can apply sequence numbers to filters. This allows you to remove filters from named and numbered ACLs without having to reconfigure an ACL.

The ability to add sequence numbers to filters simplifies updates through the ability to position a filter within an ACL. When you add a new filter, you can specify a sequence number to position the filter in the ACL and you can also remove a current filter in an ACL by specifying a sequence number.

### ACL filter sequence number behavior

- If filters with no sequence numbers are applied then the first filter is assigned a sequence number of 10, and successive filters are incremented by 10. Sequence numbers are generated automatically if they are not specified at entry.
- The maximum filter sequence number is 65535. If the sequence number exceeds this maximum, the command will not be recognized and will show the error message: `% Unrecognized command`
- If you enter a filter without a sequence number it is assigned a sequence number that is 10 greater than the last sequence number and is placed at the end of the ACL.
If you enter a filter with a sequence number which matches the sequence number on an existing filter within the same ACL, then the existing filter is overwritten with the subsequent filter.

ACL sequence numbers determine the order of execution of filters in an ACL. Filters in an ACL with a lower value sequence number are executed before filters with a higher value.

Output from the `show running-config` command displays ACL entries without filter sequence numbers. Output from relevant `show` commands displays ACL entries with their sequence numbers.

ACL sequence numbers are re-numbered upon switch restart following a `reload` command, or after powering off and powering on the switch. ACL sequence numbers are renumbered starting from 10 and increment by 10 for each filter. See the sample output in the configuration section that follows for an illustration of this behavior. No ACL sequence number re-number command is available to perform this action.

The ACL sequence number feature works with numbered and named standard and extended IPv4 and IPv6 access lists, plus named hardware IPv4 and IPv6 access lists.

The name of an access list can be designated as a number. Number in named ACLs must not exist within the range of designated numbered ACLs. (where <1-99> and <1300-1999> are standard numbered ACLs, <100-199> and <2000-2699> are extended numbered ACLs, <3000-3699> and <4000-4699> are hardware numbered ACLs).

**ACL filter sequence number applicability**

Numbered hardware ACLs are available in the range <4000-4699>, which permit or deny MAC source addresses, MAC destination addresses, and VLAN IDs to control packets coming from and going to network devices and hosts, in hardware.

**ACL filter sequence number types**

There are ACL filter sequence numbers available for the following types of ACLs:

<table>
<thead>
<tr>
<th>ACL TYPE</th>
<th>ACL COMMAND SYNTAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv4 Standard Numbered ACLs</td>
<td><code>access-list &lt;1-99&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>access-list &lt;1300-1999&gt;</code></td>
</tr>
<tr>
<td>IPv4 Extended Numbered ACLs</td>
<td><code>access-list &lt;100-199&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>access-list &lt;2000-2699&gt;</code></td>
</tr>
<tr>
<td>IPv4 Standard Named ACLs</td>
<td><code>access-list standard &lt;name&gt;</code></td>
</tr>
<tr>
<td>IPv4 Extended Named ACLs</td>
<td><code>access-list extended &lt;name&gt;</code></td>
</tr>
<tr>
<td>IPv4 Hardware Named ACLs</td>
<td><code>access-list hardware &lt;name&gt;</code></td>
</tr>
<tr>
<td>IPv6 Standard Named ACLs</td>
<td><code>ipv6 access-list standard &lt;name&gt;</code></td>
</tr>
<tr>
<td>IPv6 Extended Named ACLs</td>
<td><code>ipv6 access-list extended &lt;name&gt;</code></td>
</tr>
<tr>
<td>IPv6 Hardware Named ACLs</td>
<td><code>ipv6 access-list &lt;name&gt;</code></td>
</tr>
</tbody>
</table>
Note that ACL sequence number support for these ACL commands is optional not required. An ACL sequence number will be added automatically, starting at 10 and incrementing by 10.

**ACL commands without ACL filter sequence numbers**

ACL filter sequence numbers are not available for numbered hardware ACL commands:

**access-list <3000-3699>**

**access-list <4000-4699>**

So, when using numbered hardware ACLs, the numbered ACLs are all created individually, and applied to an interface in a series of "Access-Group" commands. The order in which the ACLs are actioned is governed by the order in which the "Access-Group" commands are configured.

But, with named hardware ACLs, the named ACL contains a series of ACL filters within it. This order in which these filters are actioned is defined by their sequence numbers.

### ACL Filter Sequence Configuration

First create a named or numbered ACL to enter ACL filters in the ACL sub-modes available:

#### Step 1: Create a new ACL and add a new filter

Create ACL 10 and then add a new filter to the access-list to permit all packets from the 192.168.1.0 subnet:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# permit 192.168.1.0 0.0.0.255
awplus(config-ip-std-acl)# end
awplus# show access-list 10
```

In the output above, you can see that, even though no sequence number was included in the command that created the filter entry, the entry has been automatically assigned the sequence number 10.
Step 2: Add another filter to the ACL

Append to, or add at the end of, ACL 10, a new filter to deny all packets from the 192.168.2 subnet:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# deny 192.168.2.0 0.0.0.255
awplus(config-ip-std-acl)# end
awplus# show access-list 10
```

In the output above, you can see that, even though no sequence number was included in the command that created the second filter entry, the entry has been automatically assigned the sequence number 20.

So, if you add a filter to an ACL without specifying a sequence number the new filter is automatically assigned a sequence number. Sequence numbers are assigned in multiples of ten from the sequence number of the last filter.

Step 3: Insert a filter into the ACL

Insert a new filter with the sequence number 15 into ACL 10 to permit packets from the 192.168.3 subnet:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# 15 permit 192.168.3.0 0.0.0.255
awplus(config-ip-std-acl)# end
awplus# show access-list 10
```

The new filter has precedence over the filter with the sequence number 20.
Step 4: Remove a filter from the ACL by specifying a filter pattern

Remove the filter with the IP address 192.168.2 from ACL 10:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# no deny 192.168.2.0 0.0.0.255
awplus(config-ip-std-acl)# end
awplus# show access-list 10
```

Step 5: Remove a filter from the ACL by specifying a sequence number

Remove the filter with the sequence number 10 from ACL 10:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# no 10
awplus(config-ip-std-acl)# end
awplus# show access-list
```

Creating ACLs in Global Configuration mode

In the case of some software ACLs, you can add new filters in Global Configuration mode with the `access-list` command. In this mode the filters are assigned a sequence number corresponding to the order in which they are entered, i.e. the first filter entered has higher precedence in the ACL.

Note - this approach to adding filters to an ACL is not available for named hardware ACLs.

Step 1: Add filters with the access-list command

Add filters to ACL 10 using the `access-list` command:

```
awplus# configure terminal
awplus(config)# access-list 10 permit 192.168.1.0 0.0.0.255
awplus(config)# access-list 10 deny 192.168.2.0 0.0.0.255
awplus(config)# end
awplus# show access-list 10
```
You can then enter the IPv4 **Standard ACL Configuration** mode and specify sequence numbers to reorder the filters, as shown in the next step.

**Step 2: Reorder the filters**

Reorder the filters in ACL 10 by specifying a sequence number for each filter. The specified sequence number will overwrite the previous sequence number assigned to the filter:

```
awplus# configure terminal
awplus(config)# access-list 10
awplus(config-ip-std-acl)# 1021 permit 192.168.1.0 0.0.0.255
awplus(config-ip-std-acl)# 3333 permit 192.168.3.0 0.0.0.255
awplus(config-ip-std-acl)# 2772 deny 192.168.2.0 0.0.0.255
awplus(config-ip-std-acl)# end
awplus# show access-list 10
```

You can see the standard IP access list 10 configured:

```
Standard IP access list 10
  10 permit 192.168.1.0, wildcard bits 0.0.0.255
  20 deny 192.168.2.0, wildcard bits 0.0.0.255
  30 permit 192.168.3.0, wildcard bits 0.0.0.255
```

**Step 3: Copy the running-config file into the startup-config file**

Copy the running-config into the file set as the current startup-config file and then reload the device. Before the reload occurs, you will receive a confirmation request saying: “reboot system? (y/n)?”.

When the device has rebooted you can then enter **Global Configuration** mode and use the **show access-list** command to display ACL 10:

```
awplus(config)# exit
awplus# copy running-config startup-config
awplus# reload
awplus# show access-list 10
```

You can see the standard IP access list 10 configured:

```
Standard IP access list 10
  10 permit 192.168.1.0, wildcard bits 0.0.0.255
  20 deny 192.168.2.0, wildcard bits 0.0.0.255
  30 permit 192.168.3.0, wildcard bits 0.0.0.255
```

After the device has rebooted, the sequence numbers of the filters in the ACL have been reassigned, incrementing from 10.
Display the ACL configuration details

Display the running system status and configuration details for ACLs:

awplus# show running-config access-list

! 
access-list 1 deny 10.1.1.0 0.0.0.255
access-list 1 permit any
access-list 2
access-list 5
access-list 10 permit 192.168.1.0 0.0.0.255
access-list 10 deny 192.168.2.0 0.0.0.255
access-list 10 permit 192.168.3.0 0.0.0.255
access-list 20
access-list 25 permit 10.1.2.0 0.0.0.255
access-list 25 deny 192.168.1.0 0.0.0.255
access-list 50
access-list 95 permit any
access-list 100
access-list 1300
access-list 2000
access-list extended acl
access-list extended my-list
access-list extended name
access-list extended name1
access-list standard name3
ipv6 access-list extended ipv6_acl
ipv6 access-list standard ipv6_acl2
ipv6 access-list extended my-ipv6-list
ipv6 access-list extended my-list
ipv6 access-list standard my-new-list
ipv6 access-list standard name
ipv6 access-list standard name1 deny any
ipv6 access-list extended name5
ipv6 access-list standard name6
access-list hw_acl
access-list icmp
access-list my-hw-list
access-list name2
access-list name4
!

Access Control Lists (ACLs) | Page 27
ACL source and destination addresses

Configure source addresses in ACL filters to filter packets coming from specified networking devices or hosts. Configure destination addresses in ACL filters to filter packets going to specified networking devices or hosts.

ACL reverse masking

ACLs uses reverse masking, also referred to as wildcard masking, to indicate to the switch whether to check or ignore corresponding IP address bits when comparing the address bits in an ACL filter to a packet being submitted to the ACL.

Reverse masking for IP address bits specify how the switch treats the corresponding IP address bits. A reverse mask is also called an inverted mask because a 1 and 0 mean the opposite of what they mean in a subnet or a network mask.

- A reverse mask bit 0 means check the corresponding bit value.
- A reverse mask bit 1 means ignore the corresponding bit value.