



IPv6 Access Control

bintec Dm808-I

Copyright© Version 11.09 bintec elmeg

Legal Notice

Warranty

This publication is subject to change.

bintec offers no warranty whatsoever for information contained in this manual.

bintec is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

I	Related Documents	1
Chapter 1	Introduction	2
1.1	IPv6 Access Control Lists	2
Chapter 2	Configuration	4
2.1	Accessing the Configuration	4
2.2	Main Configuration Menu	4
2.2.1	? (HELP)	4
2.2.2	ACCESS-LIST	4
2.2.3	NO	5
2.2.4	STATEFUL	5
2.2.5	EXIT	6
2.3	IPv6 Stateless Access Control Lists	6
2.3.1	? (HELP)	6
2.3.2	DESCRIPTION	7
2.3.3	ENTRY	7
2.3.4	NO	11
2.3.5	EXIT	12
2.4	IPv6 Stateful Access Control Lists	12
2.4.1	? (HELP)	12
2.4.2	DESCRIPTION	12
2.4.3	ENTRY	13
2.4.4	NO	22
2.4.5	EXIT	23
2.5	Associating an access control list to the IPv6 protocol	23
2.6	Associating an IPv6 access control list to an interface	24
Chapter 3	Monitoring	25
3.1	Monitoring Commands	25
3.1.1	LIST	25
Chapter 4	Configuration Examples	28
4.1	Scenario 1	28
Chapter 5	Annex	31
5.1	Reserved Ports	31
5.2	Reserved Protocols	31

I Related Documents

bintec Dm715-I BRS

bintec Dm745-I Policy Routing

bintec Dm752-I Access Control

bintec Dm764-I Route Mapping

bintec Dm786-I AFS

Chapter 1 Introduction

1.1 IPv6 Access Control Lists

Routers use access control lists (ACL) to filter traffic transmitted through them. This manual focuses on IPv6 access control lists. IPv4 access control lists are explained in manual bintec Dm752-I Access Control.

IPv6 access control lists allow you to filter the flow of IPv6 packets transmitted through router interfaces. An IPv6 access control list is made up of a series of entries that define the properties a packet must have to belong to a certain group. Each entry in said list contains a series of requirements and an action identified by a unique number (entry identifier or ID field). This can be seen in Figure 1.

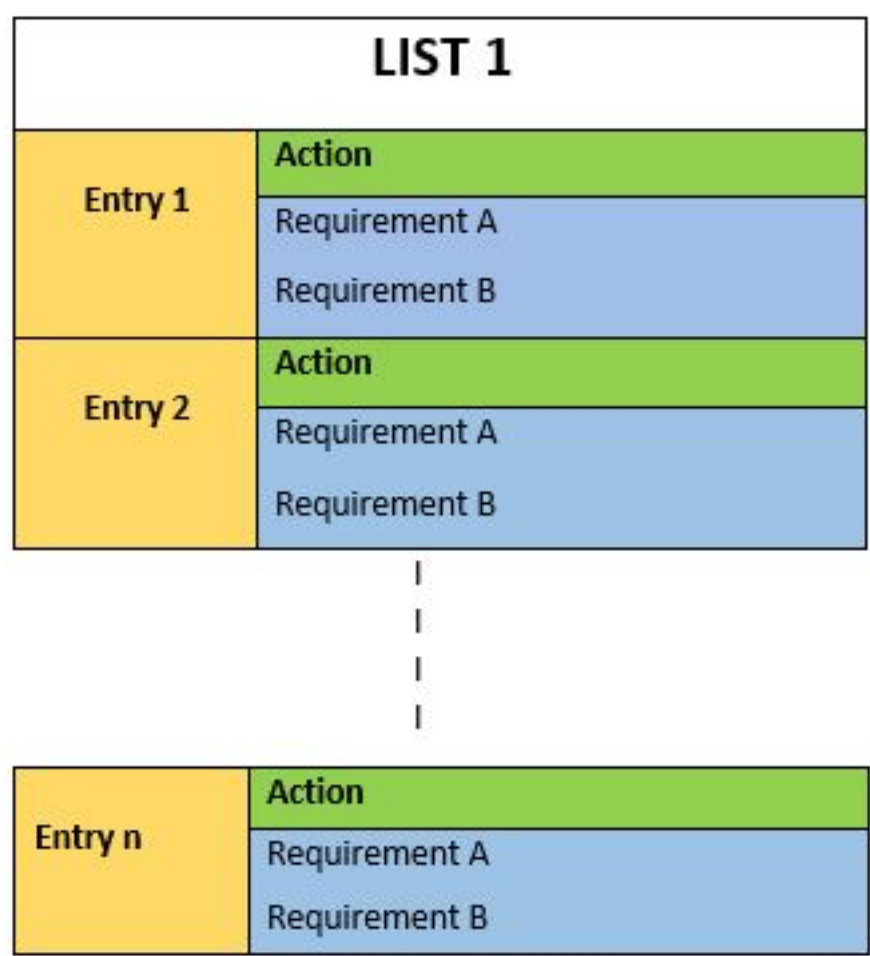


Fig. 1: Access control list with n entries

Requirements can be made up of a source IP address (or range of addresses), a destination IP address (or range of destination IP addresses), a protocol (or range of protocols), source and destination ports (or range of ports), DSCP (*Differentiated Service Code Point*), label (assigned to IPv6 in the classification process), a pre-established tcp session, etc. Only relevant elements need to be defined. Said action determines the process applied to IPv6 packets matching the criteria for the entry's requirements. This action can be one of two types: *Permit* or *Deny*. (See Figure 2).

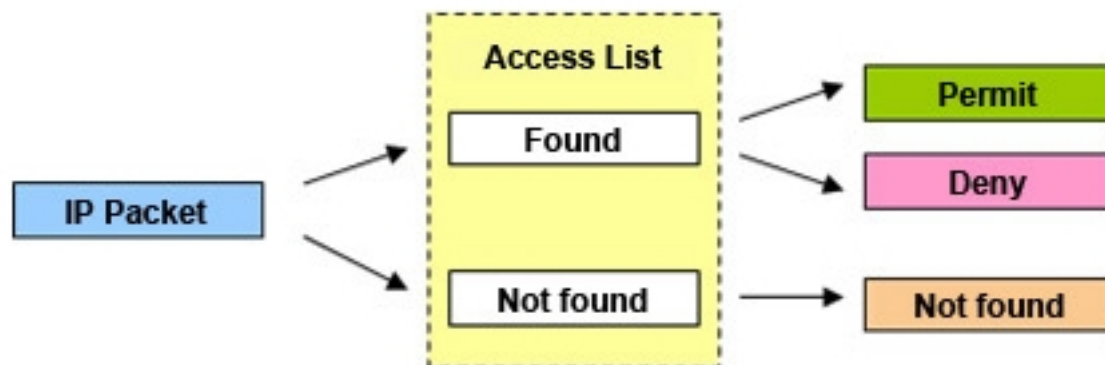


Fig. 2: Actions over the IPv6 packet

An IPv6 access control list is not a filter to limit packet flow in the router. IPv6 access control lists must be associated to an interface where IPv6 is enabled or, globally, to the IPv6 protocol. Additionally, you need to decide whether the filter is going to be applied to input traffic (avoiding router overload) and/or to output traffic. When you globally configure a filter in the IPv6 protocol, this filter is only applied to local traffic (i.e. traffic that is locally generated or directed to the device).

An access list can also be associated to a different Access Control function, such as bandwidth reservation or Route Mapping. For further information on the use of access lists for this feature, please see manuals bintec Dm715-I BRS and bintec Dm764-I Route Mapping.

Types of access lists:

Stateless: check each packet's source and destination address. They can also verify specific protocols, port numbers and other parameters.

Stateful: check both the source and destination address of the packet, as well as the state and type of session. To configure stateful lists, the AFS feature must be enabled (please see manual bintec Dm786-I AFS).

Chapter 2 Configuration

2.1 Accessing the Configuration

Operations to create, view, modify or eliminate IPv6 access control lists are executed from a specific menu.

In the router configuration structure, IPv6 Access Controls are organized as a *feature*. To view the features that allow you to configure the router, enter the **feature** command followed by a question mark (?).

To access the IPv6 access controls configuration menu, enter **feature** from the configuration root menu followed by **ipv6-access-list**.

Example:

```
Config>feature ipv6-access-list
-- IPv6 Access Lists user configuration --
IPv6 Access Lists config>
```

The main IPv6 access controls feature configuration menu is then accessed. Here, you can create, eliminate and view IPv6 access control lists.

Each access control list is made up of entries that set criteria and parameters allowing or restricting access.

2.2 Main Configuration Menu

Lists can be created and eliminated from the main IPv6 access control configuration menu.

If an entry parameter or option isn't configured in the access control list, it is ignored when checking said access.

The order of the entries in an access control list is very important when requirements apply to different entries. Please note that the order of the entries in the list is not defined by the entry number, but depends on the order they were entered in. If (when going through the list) an element in the first listed entry matches the application search, the search stops and the prescribed action is carried out.

The following commands are found in the main IPv6 access control menu:

Command	Function
? (HELP)	Lists the available commands or their options.
ACCESS-LIST	Configures an access list.
NO	Negates a command or sets the default value.
STATEFUL	Configures a stateful access list.

2.2.1 ? (HELP)

Lists the valid commands at the level in which the router is programmed. You can also enter this after a specific command to list the available options.

Syntax:

```
IPv6 Access Lists config>?
```

Example:

```
IPv6 Access Lists config>?
access-list    Configure an IPv6 access-list
no             Negate a command or set its defaults
stateful       Configure an IPv6 stateful access list
exit
IPv6 Access Lists config>
```

2.2.2 ACCESS-LIST

Creates an IPv6 stateless access control list and accesses the submenu to configure the entries in said list. IPv6 access control lists are identified via a name selected by the user.

On introducing this command followed by an identifier, you access a submenu where said access list can be configured. The list created appears at the new prompt:

Syntax:

```
IPv6 Access Lists config>access-list ?  
  <1..50 chars>      IPv6 access-list name  
IPv6 Access Lists config>
```

Example:

```
IPv6 Access Lists config>access-list list1  
IPv6 Access List list1>
```

Each stateless access list must have a unique name. If you try to configure one using a name that already exists, the router returns an error message.

Example:

```
IPv6 Access Lists config>access-list list2  
CLI Error: Stateful access-list with name list2 already exists, delete it first  
CLI Error: Command error  
IPv6 Access Lists config>
```

2.2.3 NO

Disables features or sets the default values in some parameters.

Syntax:

```
IPv6 Access Lists config>no ?  
  access-list      Configure an IPv6 access-list  
  stateful         Configure an IPv6 stateful access list  
IPv6 Access Lists config>
```

2.2.3.1 NO ACCESS-LIST

Eliminates an IPv6 access control list.

Syntax:

```
IPv6 Access Lists config>no access-list ?  
  <1..50 chars>      IPv6 access-list name  
IPv6 Access Lists config>
```

Example:

```
IPv6 Access Lists config>no access-list list1  
IPv6 Access Lists config>
```

2.2.3.2 NO STATEFUL ACCESS-LIST

Eliminates an IPv6 stateful access list.

Syntax:

```
IPv6 Access Lists config>no stateful ?  
  access-list      Set IPv6 access list name  
IPv6 Access Lists config>no stateful access-list ?  
  <1..50 chars>      IPv6 access list name  
IPv6 Access Lists config>
```

Example:

```
IPv6 Access Lists config>no stateful access-list list2  
IPv6 Access Lists config>
```

2.2.4 STATEFUL

Creates an IPv6 stateful access control list and accesses the submenu where you can configure entries in said list. IPv6 stateful access control lists are identified by a name selected by the user.

When you enter this command followed by an identifier, you access a submenu where you can configure said access list. The list created appears at the new prompt.

Syntax:

```
IPV6 Access Lists config>stateful ?
  access-list      Set IPv6 access list name
IPV6 Access Lists config>stateful access-list ?
  <1..50 chars>    IPv6 access list name
IPV6 Access Lists config>
```

Example:

```
IPV6 Access Lists config>stateful access-list list2
Stateful IPv6 Access List list2>
```

The Advanced Firewall System (AFS) feature must be enabled to configure a stateful access list. Otherwise, an error message is displayed:

```
IPV6 Access Lists config>stateful access-list list2
CLI Error: You must enable afs to configure stateful access lists
CLI Error: Command error
IPV6 Access Lists config>
```

Each stateful access list must have a unique name. If you try and configure one with a name that already exists, the router returns an error message.

Example:

```
IPV6 Access Lists config>stateful access-list list1
CLI Error: Stateless access-list with name list1 already exists, delete it first
CLI Error: Command error
IPV6 Access Lists config>
```

2.2.5 EXIT

Exits the IPv6 access controls feature configuration environment and returns to the general configuration prompt.

Syntax:

```
IPV6 Access Lists config>exit
```

Example:

```
IPV6 Access Lists config>exit
Config>
```

2.3 IPv6 Stateless Access Control Lists

Access this menu to create an IPv6 stateless access control list.

Example:

```
IPV6 Access Lists config>access-list list1
IPv6 Access List list1>
```

The IPv6 access control list submenu accepts the following subcommands:

Command	Function
? (HELP)	Lists the available commands or their options.
DESCRIPTION	Inserts a textual description of an access control list.
ENTRY	Configures an entry for this access-list.
NO	Negates a command or sets its default value.

2.3.1 ? (HELP)

Lists the commands available at the level in which the router is programmed. You can also enter this after a specific command to list the available options.

Syntax:

```
IPV6 Access List listID>?
```

Example:

```
IPV6 Access List list1>
  description      Configure a description for this access list
  entry            Configure an entry of this access-list
  no               Negates a command or sets its defaults
  exit
IPV6 Access List list1>
```

2.3.2 DESCRIPTION

Inserts a text description in an access control list. This will help you understand its purpose or function later on.

Syntax:

```
IPV6 Access List listID>description ?
  <word>      Description text
IPV6 Access List listID>
```

Example:

```
IPV6 Access List list1>description "Description example"
```

2.3.3 ENTRY

Creates or modifies an entry in an access control list. This command must always be entered with the entry identifier.

A new entry is created every time you enter this command, followed by a unique identifier (i.e., one that doesn't already exist). If you enter one that does, the value will be modified.

Syntax:

```
IPV6 Access List listID>entry<id><parameter> [value]
```

Configuration options for an entry are as follows:

```
IPV6 Access List listID>entry <id> ?
  deny            Configures type of entry or access control as deny
  description      Sets a description for the current entry
  destination      Destination menu: subnet or port
  dscp             IPv6 header dscp field
  label           Label for classification
  no              Negate a command or set its defaults
  permit          Configures type of entry or access control as permit
  protocol         Protocol
  protocol-range   Protocol range
  source           Source menu: subnet or port
  tcp-specific     Tcp specific filtering
IPV6 Access List listID>
```

2.3.3.1 ENTRY <id> DENY

Identifies the entry as DENY. Said parameter indicates that the matching criteria for IPv6 traffic (set as requirements) is denied. This command is an **action** indicator and sets the action for IPv6 packets matching the entry as DENY.

Syntax:

```
IPV6 Access List listID>entry <id> deny
```

Example:

```
IPV6 Access List list1>entry 3 deny
IPV6 Access List list1>
```

2.3.3.2 ENTRY <id> DESCRIPTION

Adds a text description to an entry (to help understand it better).

Syntax:

```
IPV6 Access List listID>entry <id> description ?
    <word>      Description text
IPV6 Access List listID>
```

Example:

```
IPV6 Access List list1>entry 1 description "first entry"
IPV6 Access List list1>
```

2.3.3.3 ENTRY <id> DESTINATION

Configures a range of destination IPv6 addresses or a range of destination ports affected by the entry. This command is a **requirement** indicator.

Syntax:

```
IPV6 Access List listID>entry <id> destination ?
    Address IPv6 address and prefix of the destination subnet
    Port      Destination port range
```

2.3.3.3.1 ENTRY <id> DESTINATION ADDRESS

Establishes a range of destination IPv6 addresses affected by the entry. This is indicated through an IPv6 prefix.

Syntax:

```
IPV6 Access List listID>entry <id> destination address ?
    <a::b/l>      Ipv6 prefix
IPV6 Access List listID>
```

Example:

```
IPV6 Access List list1>entry 3 destination address 2001:db8:a::/85
IPV6 Access List list1>
```

2.3.3.3.2 ENTRY <id> DESTINATION PORT

Allows or denies access to various TCP or UDP destination ports.

These must be followed by two numbers. This is a range of ports affecting the entry, the first being the lower limit and the second, the upper. If no range is required, simply enter two equal values. Port values go from 0 to 65535.

Syntax:

```
IPV6 Access List listID>entry <id> destination port<lower_limit> <higher_limit>
```

Example:

```
IPV6 Access List list1>entry 3 destination port 2 4
IPV6 Access List list1>
```

This entry matches all TCP or UDP packets whose destination port is between 2 and 4 (inclusive).

2.3.3.4 ENTRY <id> DSCP

Sets the IPv6 packets affected by the entry based on the value of the DSCP field (*Differentiated Services Code Point*). This field is found in the IPv6 header and establishes the type of service the IPv6 packet requires. The DSCP field can take values between 0 and 63. This command is a **requirement** indicator.

Syntax:

```
IPV6 Access List listID>entry <id> dscp ?
    <0..63>      Value in the specified range
IPV6 Access List listID>
```

Example:

```
IPV6 Access List list1>entry 3 dscp 12
IPV6 Access List list1>
```

2.3.3.5 ENTRY <id> LABEL

IPv6 packets are affected by the entry through its *label*, which is an internal parameter associated to said packets. This is a number between 0 and 99, used to select, classify and filter IPv6 traffic.

By default, all IPv6 packets have a label value equal to 0. This value can be changed through Policy Routing (see manual bintec Dm745-I Policy Routing), using an appropriately configured Route Map (see manual bintec Dm764-I Route Mapping). Traffic marked with a label can be subsequently selected from an access list through **entry <id> label**. This command is a **requirement** indicator.

Syntax:

```
IPv6 Access List listID>entry <id> label ?
    <0..99>      Label value
IPv6 Access List listID>
```

Example:

```
IPv6 Access List list1>entry 3 label 12
IPv6 Access List list1>
```

2.3.3.6 ENTRY <id> PERMIT

Identifies the entry as PERMIT. Said parameter indicates all IPv6 traffic matching criteria (set as requirements) is PERMITTED. This command is an **action** indicator and determines the action for IPv6 packets matching the entry is PERMIT.

Syntax:

```
IPv6 Access List listID>entry <id> permit
```

Example:

```
IPv6 Access List list1>entry 3 permit
IPv6 Access List list1>
```

2.3.3.7 ENTRY <id> PROTOCOL

Establishes what protocol is affected by the entry. This command must be followed by the protocol number (value between 0 and 255, see annex 2) or the name. If you specify IP protocol, any protocol is admitted. If you specify ICMP protocol, you can specify its type and code as well (either by number or by name).

The aim of this command is to allow or deny access to various protocols. This command is a **requirement** indicator.

Syntax:

```
IPv6 Access List listID>entry <id> protocol ?
    <0..255>      An IP protocol number
    esp          Encapsulation Security Payload
    icmp         Internet Control Message Protocol
    ip           Any Internet Protocol
    tcp          Transmission Control Protocol
    udp          User Datagram Protocol
IPv6 Access List listID>entry <id> protocol icmp ?
    <0..255>      ICMPv6 message type
    beyond-scope Destination beyond scope
    destination-unreachable Destination address is unreachable
    dhaad-reply   Home agent address discovery reply
    dhaad-request Home agent address discovery request
    echo-reply    Echo reply
    echo-request  Echo request (ping)
    header        Parameter header problems
    hop-limit     Hop limit exceeded in transit
    mld-done      Multicast Listener Discovery Done
    mld-query     Multicast Listener Discovery Query
    mld-report    Multicast Listener Discovery Report
    mpd-advertisement Mobile prefix advertisement
    mpd-solicitation Mobile prefix solicitation
    nd-na         Neighbor discovery neighbor advertisements
    nd-ns         Neighbor discovery neighbor solicitations
    next-header   Parameter next header problems
```

```

no-admin           Administration prohibited destination
no-route           No route to destination
packet-too-big     Packet too big
parameter-option   Parameter option problems
parameter-problem  All parameter problems
port-unreachable   Port unreachable
reassemble-timeout Reassembly timeout
redirect           Neighbor redirect
reject-route       Reject route to destination
renum-command      Router renumbering command
renum-result       Router renumbering result
renum-seq-number   Router renumbering sequence number reset
router-advertisement Neighbor discovery router advertisements
router-renumbering All router renumbering
router-solicitation Neighbor discovery router solicitations
source-policy      Source address failed ingress/egress policy
time-exceeded      All time exceeded
unreachable        All unreachable
<cr>
IPv6 Access List listID>entry <id> protocol icmp <type> ?
<0..255>          ICMPv6 message code
<cr>

```

Example:

```

IPv6 Access List list1>entry 3 protocol icmp
IPv6 Access List list1>

```

Command history:

Release

11.01.09

Modification

ICMP type and code options were introduced as of version 11.01.09.

2.3.3.8 ENTRY <id> PROTOCOL-RANGE

Establishes a range of protocols affected by the entry. This command must be followed by two numbers. The first corresponds to the protocol identifier in the lower range, the second to the upper one. If no range is required, enter two equal values. Both protocol identifiers must be between 0 and 255.

The aim of this command is to allow or deny access to various protocols. This command is a **requirement** indicator.

Syntax:

```

IPv6 Access List listID>entry <id> protocol-range <lower_prot> <higher_prot>

```

Example:

```

IPv6 Access List list1>entry <id> protocol-range 21 44
IPv6 Access List list1>

```

2.3.3.9 ENTRY <id>SOURCE

Configures a range of source IPv6 addresses or a range of source ports affected by the entry. This command is a **requirement** indicator.

Syntax:

```

IPv6 Access List listID>entry <id> source ?
  Address  IPv6 address and prefix of the source subnet
  port     Source port range
IPv6 Access List listID>

```

2.3.3.9.1 ENTRY <id> SOURCE ADDRESS

Establishes the affected range of source IPv6 addresses. This is an IPv6 prefix.

Syntax:

```

IPv6 Access List listID>entry <id> source address ?
<a::b/l>    Ipv6 prefix
IPv6 Access List listID>

```

Example:

```
IPV6 Access List list1>entry 3 source address 2001:db8::/64
IPV6 Access List list>
```

2.3.3.9.2 ENTRY <id> SOURCE PORT

Allows or denies access to various TCP or UDP source ports. These must be followed by two numbers linked to the ports that affect the entry (the first number represents the lower limit, while the second one represents the upper limit). Enter two equal values if you chose not to have a range. Port values must be between 0 and 65535.

Syntax:

```
IPV6 Access List listID>entry <id> source port <lower_limit> <higher_limit>
```

Example 1:

```
IPV6 Access List list1>entry 3 source port 2 4
IPV6 Access Listlist1>
```

This entry matches all TCP or UDP packets whose source port is between 2 and 4 (inclusive).

2.3.3.10 ENTRY <id> TCP-SPECIFIC ESTABLISHED

Establishes the access control requirement for IPv6 TCP packets, depending on whether the TCP session has been pre-established or not. To find out if a TCP session is already established, check that the ACK or the RST bit in the TCP packet header is present. If one of the two is there, then the session is considered established. This command is a **requirement** indicator.

Syntax:

```
IPV6 Access List listID>entry <id> tcp-specific ?
    established-state    Established tcp connections
IPV6 Access List listID>
```

Example:

```
entry 1 permit
entry 1 protocol tcp
entry 1 tcp-specific established-state
```

2.3.4 NO

Disables functionalities or sets the default values in some parameters.

Syntax:

```
IPV6 Access List listID>no ?
    description    Configure a description for this access list
    entry          Configure an entry of this access -list
```

2.3.4.1 NO DESCRIPTION

Eliminates a textual description associated to an IPv6 access control list.

Syntax:

```
IPV6 Access List listID>no description
```

Example:

```
IPV6 Access List list1>no description
IPV6 Access List list1>
```

2.3.4.2 NO ENTRY

Eliminates an entry from the IPv6 access control list. Enter the identifier for the entry you wish to remove.

Syntax:

```
IPV6 Access List listID>no entry <id>
```

Example:

```
IPv6 Access List list1>no entry 3
IPv6 Access List list1>
```

2.3.5 EXIT

Exits the configuration environment for an IPv6 access control list and returns to the main IPv6 access control menu prompt.

Syntax:

```
IPv6 Access List listID>exit
```

Example:

```
IPv6 Access List list1>exit
IPv6 Access Lists config>
```

2.4 IPv6 Stateful Access Control Lists

Access this menu when an IPv6 stateful access control list is created.

At the new submenu prompt, indicate the list is Stateful and its identifier.

To configure these access lists, the AFS feature must be enabled. If you are going to execute a dynamic change while the session is active, bear in mind you will need to end the session before the change takes effect. To do this, disable and enable the AFS feature using the **no enable** and **enable** commands found in the AFS configuration menu (please see manual bintec Dm786-I AFS).

Example:

```
IPv6 Access Lists config>stateful access-list list2

Stateful IPv6 Access List list2>
```

The IPv6 access control submenu accepts the following subcommands:

Command	Function
? (HELP)	Lists the commands or their available options.
DESCRIPTION	Inserts a text description for an access control list.
ENTRY	Configures an entry for this access-list.
NO	Negates a command or sets its default value.

2.4.1 ? (HELP)

Lists the valid commands at the layer the router is programmed at. You can also use this command after a specific command to list the available options.

Syntax:

```
Stateful IPv6 Access List listName>?
```

Example:

```
Stateful IPv6 Access List list2>?
description      Description of this rule
entry            Configure an entry for this access list
no               Negate a command or set its defaults
exit
Stateful IPv6 Access List list2>
```

2.4.2 DESCRIPTION

Inserts a text description for an access control list. This will help you to understand its purpose or function in the future.

Syntax:

```
Stateful IPv6 Access List listName>description ?
```



```
<word>      Text
Stateful IPv6 Access List listName>
```

Example:

```
Stateful IPv6 Access List list2>description "Description example"
```

2.4.3 ENTRY

Creates or modifies an entry on an access control list. This command must always be followed by the entry identifier.

A new entry is created each time you enter this command, followed by an identifier that is not already in the list. When you enter an already existing identifier, this modifies the value of the entered parameter.



Note

Unlike what happens with generic or stateless access control lists, it's possible to configure more than one selection criterion for the same entry (as long as the packets fulfill ALL of the selection criteria specified for the entry in order for it to match).

This can be very useful when you want to match packets that do not fulfill several criteria simultaneously (e.g. when you want the destination address to be one that is not listed).

Syntax:

```
Stateful IPv6 Access List listName>entry <id> <parameter> [value]
```

Configuration options for a Stateful entry are as follows:

Syntax:

```
Stateful IPv6 Access List listName>entry <id> ?
  ah                Authentication header
  app-detect        Match on app-detect information
  app-id            Match on session app-id
  conn-limit        Match an specific connection limit
  default           Set default values
  deny             Deny this entry
  description       Entry description
  destination       Destination match criteria
  dscp-field        DSCP in IP packets
  header            Header type filter
  hex-string        Search an specific hexadecimal string
  in-interface      Match an incoming interface
  label            Label for classification
  length-interval   Define a datagram length interval to match to
  no               Negate the following match criteria
  out-interface     Match an outgoing interface
  permit           Permit this entry
  protocol          IP protocol matching options
  protocol-range    Specify a protocol range
  rate-limit        Match an specific rate limit in kbps
  rtp              Match any RTP packet flow
  session          Define a session control match
  source           Source match criteria
  string           Search an specific string
  tcp-flags        Match an specific tcp flag
  webstr           Filter urls/hosts in http sessions
  webserv          Filter urls in http sessions
Stateful IPv6 Access List listName>
```

2.4.3.1 ENTRY <id> AH

Selects a packet depending on the IPSec packets authentication header parameters. These parameters take the SPI value, for which a range is entered, and (optionally) the header length.

Syntax:

```
Stateful IPv6 Access List listName>entry <id> ah <minSPI> <maxSPI> [length]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 ah 0 ffff
Stateful IPV6 Access List list2>
```

2.4.3.2 ENTRY <id> APP-DETECT HOST

Matches the session host drawn by AFS's **app-detect** feature with the regular expression given. Any session where a host is detected (HTTP Host, Referer (host-only) or SSL Host) is tried for a match. If no session host is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect host <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect host "googlevideo\.com"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.3 ENTRY <id> APP-DETECT HTTP-HOST

Matches the HTTP Host session drawn by AFS's **app-detect** feature with the regular expression given. If no HTTP Host session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect http-host <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect http-host "ebay\.com"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.4 ENTRY <id> APP-DETECT HTTP-REFERER

Matches the HTTP Referer session drawn by AFS's **app-detect** feature with the regular expression given. If no HTTP Referer session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect http-referer <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect http-referer "ebay\.com"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.5 ENTRY <id> APP-DETECT HTTP-URL

Matches the HTTP URL session drawn by AFS's **app-detect** feature with the regular expression given. If no HTTP URL session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect http-url <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect http-url "motors"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.6 ENTRY <id> APP-DETECT HTTP-USER-AGENT

Matches the HTTP User-agent session drawn by AFS's **app-detect** feature with the regular expression given. If no HTTP User-agent session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect http-user-agent <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect http-user-agent "Chrome"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.7 ENTRY <id> APP-DETECT SSL-HOST

Matches the SSL server hostname session drawn by AFS's **app-detect** feature with the regular expression given. If no SSL hostname session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect ssl-host <1..150 chars>
```

Example:

```
Stateful Access List 5000> entry 1 app-detect ssl-host "googlevideo\..com"
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.8 ENTRY <id> APP-DETECT SSL

Matches the SSL sessions detected by **app-detect**. AFS's **app-detect** feature must be configured for the command to work. If no SSL session is detected when **app-detect** is configured, there is no match.

Syntax:

```
Stateful Access List #>entry <id> app-detect ssl
```

Example:

```
Stateful Access List 5000> entry 1 app-detect ssl
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.9 ENTRY <id> APP-ID

Matches the AFS session AppId.

Syntax:

```
Stateful Access List #>entry <id> app-id ?
  13 protocol-number <0..255>    Match on layer 3 (protocol)
  14 port-number <0..65535>      Match on layer 4 (port)
  custom id <0..65535>           Match on custom app-id
```

Example:

```
Stateful Access List 5000> entry 1 app-id 14 port-number 80
```

Command history:

Release	Modification
11.01.01	This command was introduced as of version 11.01.01.

2.4.3.10 ENTRY <id> CONN-LIMIT

Sets, through the mask length, a connection limit for a given IP address or a certain group of hosts. Any packet with a value above this limit is considered to match this criteria.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> conn-limit <limit> <mask>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 conn-limit 3 64
Stateful IPV6 Access List list2>
```

2.4.3.11 ENTRY <id> DEFAULT

Sets all parameters belonging to a Stateful entry to their default values.

These are:

- PERMITTED
- ADDRESS: ::/0

Syntax:

```
Stateful IPV6 Access List listName>entry <id> default
```

Example:

```
Stateful IPV6 Access List list2>entry 1 default
Stateful IPV6 Access List list2>
```

2.4.3.12 ENTRY <id> DENY

Identifies the entry as DENY. Therefore, the IPv6 traffic that meets the criteria set in the requirements is NOT permitted (i.e. it cannot get through the access control). Since this command is an **action** indicator, IPv6 packets that match the entry CANNOT pass.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> deny
```

Example:

```
Stateful IPV6 Access List list2>entry 1 deny
Stateful IPV6 Access List list2>
```

2.4.3.13 ENTRY <id> DESCRIPTION

Adds a text description to better understand the entry's purpose, or for later use.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> description <description>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 description "Access list example"
Stateful IPV6 Access List list2>
```

2.4.3.14 ENTRY <id> DESTINATION

Selects a packet based on different destination parameters, e.g. destination IP address or port (tcp or udp).

Syntax:

```
Stateful IPV6 Access List listName>entry <id> destination ?
  address      Destination IP network
  tcp          Match tcp protocol
  udp          Match udp protocol
```

```
Stateful IPV6 Access List listName>
```

2.4.3.14.1 ENTRY <id> DESTINATION ADDRESS

Selects a packet depending on its destination IP. This is indicated through an IPv6 prefix.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> destination address <ipv6Prefix>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 destination address 2001:db8::/64
Stateful IPV6 Access List list2>
```

2.4.3.14.2 ENTRY <id> DESTINATION TCP PORT

Specifies a port or a range of TCP destination ports. The packet must be TCP to match this criteria.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> destination tcp port <low-port> [<high-port>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 destination tcp port 20000 21000
Stateful IPV6 Access List list2>
```

2.4.3.14.3 ENTRY <id> DESTINATION UDP PORT

Specifies a port or a range of UDP destination ports. The packet must be UDP to match this criteria.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> destination udp port <low-port> [<high-port>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 destination udp port 20000 21000
Stateful IPV6 Access List list2>
```

2.4.3.15 ENTRY <id> DSCP-FIELD

Specifies the value of the dscp field for the IP packet Type of Service byte. Values can range from 0 to 63.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> dscp-field <value>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 dscp-field 33
Stateful IPV6 Access List list2>
```

Command history:

Release	Modification
11.01.06	This command was introduced as of version 11.01.06.

2.4.3.16 ENTRY <id> HEADER

Selects a packet based on whether certain extension headers are found in the IPv6 packet.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> header {hop | routing | fragment | auth | dest}
```

Example:

```
Stateful IPV6 Access List list2>entry 1 header fragment
Stateful IPV6 Access List list2>
```

2.4.3.17 ENTRY <id> HEX-STRING

Specifies a hexadecimal string. The AFS system inspects the packet looking for this string. If found, the packet is considered matching.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> hex-string <string>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 hex-string AABBC
Stateful IPV6 Access List list2>
```

2.4.3.18 ENTRY <id>IN-INTERFACE

Specifies an **in-interface**.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> in-interface <interface>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 in-interface ethernet0/0
Stateful IPV6 Access List list2>
```

2.4.3.19 ENTRY <id> LABEL

IPv6 packets are affected by the entry through its *label*, which is an internal parameter associated to said packets. This is a number between 0 and 99, used to select, classify and filter IPv6 traffic.

By default, all IPv6 packets have a label value equal to 0. This value can be changed through Policy Routing (see manual bintec Dm745-I Policy Routing), using an appropriately configured Route Map (see manual bintec Dm764-I Route Mapping). Traffic marked with a label can be subsequently selected in an access list (**entry <id> label** command).

Syntax:

```
Stateful IPV6 Access List listName>entry <id> label <label-value> [mask <label-mask>]
```

The values to configure are as follows:

<i>id</i>	The entry identifier to be configured.
<i>label-value</i>	Value the packet label must take.
<i>label-mask</i>	Mask specifying what packet label bits are going to be checked.

Example:

```
Stateful IPV6 Access List list2>entry 1 label 1
Stateful IPV6 Access List list2>
```

Command history:

Release	Modification
11.01.06	This command was introduced as of version 11.01.06.

2.4.3.20 ENTRY <id>LENGTH INTERVAL

Specifies a length interval for a packet. A packet is considered matching if its length falls within the interval.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> length-interval <low> <high>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 length-interval 1000 5000
Stateful IPV6 Access List list2>
```

2.4.3.21 ENTRY <id> NO

If you select the no option in front of the selection criteria, a packet is considered as matching if it **DOESN'T** fulfill the selection criteria specified.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> no <criteria>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 no length-interval 1000 1500
Stateful IPV6 Access List list2>
```

2.4.3.22 ENTRY <id> OUT-INTERFACE

Specifies an **out-interface**.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> out-interface <interface>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 out-interface ethernet0/0
Stateful IPV6 Access List list2>
```

2.4.3.23 ENTRY <id> PERMIT

Identifies the entry as PERMIT. Therefore, all IPv6 traffic that meets the criteria set in the requirements can pass through the access control list. Since this command is an **action** indicator, IPv6 packets that match the entry can pass.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> permit
```

Example:

```
Stateful IPV6 Access List list2>entry 1 permit
Stateful IPV6 Access List list2>
```

2.4.3.24 ENTRY <id> PROTOCOL

Selects a packet depending on the IP-encapsulated protocol.

The list of protocols supported in this command appears in the Annex below.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> protocol <protocol>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 protocol tcp
Stateful IPV6 Access List list2>
```

Some of the selected protocols can admit suboptions such as **peer2peer**.

```
Stateful IPV6 Access List listName>entry 1 protocol peer2peer ?
all          All peer to peer traffic
apple        AppleJuice traffic
ares         Ares AresLite traffic
bit-torrent  BitTorrent traffic
dc           Direct Connect traffic
e-mule       E-mule E-donkey traffic
gnutella     Gnutella traffic
kazaa        Kazaa traffic
mute         Mute traffic
soul         SoulSeek traffic
waste        Waste traffic
winmx        WinMx traffic
xdcc         XDCC traffic
```

Example:

```
Stateful IPV6 Access List list2>entry 1 protocol peer2peer all
Stateful IPV6 Access List list2>
```

2.4.3.25 ENTRY <id> PROTOCOL-RANGE

Selects a packet based on a range of IP protocols. The range is specified using the numerical values of protocols.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> protocol-range <limit1> <limit2>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 protocol-range 1 17
Stateful IPV6 Access List list2>
```

2.4.3.26 ENTRY <id> RATE-LIMIT

Specifies a limit in kilobits per second. When this is surpassed, the packet is considered to match this criteria. This is measured in kilobits per second.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> rate-limit <limit> [<burst>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 rate-limit 100
Stateful IPV6 Access List list2>
```

2.4.3.27 ENTRY <id> RTP

UDP flows are automatically screened for RTP traffic. If a packet pertains to a flow classified as RTP, it is considered to match this criteria. You can also filter based on the type of traffic transported by RTP (audio, video) or by the **payload-type** defined.

Syntax:

```
Stateful Access List listName>entry <id> rtp [audio | video | payload-type <type>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 rtp
Stateful IPV6 Access List list2>
```

2.4.3.28 ENTRY <id> SESSION

The selection criteria depend on the session:

Syntax:

```
Stateful IPV6 Access List listName>entry <id> session ?
    expire      Match a session expire time interval
    state       Match an specific session state
Stateful IPV6 Access List listName>
```

2.4.3.28.1 ENTRY <id> SESSION EXPIRE

The selection criteria is the lifetime the session has left. This command specifies a time interval.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> session expire <low> <high>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 session expire 500 1000
Stateful IPV6 Access List list2>
```


2.4.3.28.2 ENTRY <id> SESSION STATE

The selection criteria is the session's state (i.e. if it is new, already established, executing source or destination NAT).

Syntax:

```
Stateful IPV6 Access List listName>entry <id> session state <state>
```

The possible states for a session are as follows:

<i>invalid</i>	The session is in an invalid state, ready to be deleted.
<i>new</i>	The session is new, this is the first packet for this session.
<i>established</i>	The session is established.
<i>awaited</i>	The session is expected by an ALG.
<i>untrack</i>	The IP packet doesn't have a session. It couldn't be created.
<i>source-nat</i>	NAT is applied at session source.
<i>destination-nat</i>	NAT is applied at session destination.
<i>app-detecting</i>	The application detection feature is working on the session.

Example:

```
Stateful IPV6 Access List list2>entry 1 session state established
Stateful IPV6 Access List list2>
```

2.4.3.29 ENTRY <id> SOURCE

Selects a packet based on its source parameters, e.g. source IP address or source port (tcp or udp).

Syntax:

```
Stateful IPV6 Access List listName>entry <id> source ?
  address      Source IP network
  tcp          Match tcp protocol
  udp          Match udp protocol
Stateful IPV6 Access List listName>
```

2.4.3.29.1 ENTRY <id> SOURCE ADDRESS

Selects a packet based on its source IP. This is indicated through an IPv6 prefix.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> source address <ipv6Prefix>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 source address 2001:db8::/64
Stateful IPV6 Access List list2>
```

2.4.3.29.2 ENTRY <id> SOURCE TCP PORT

Specifies a port or a range of TCP source ports. The packet must be TCP to match this criteria.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> source tcp port <low-port> [<high-port>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 source tcp port 10000 12000
Stateful IPV6 Access List list2>
```

2.4.3.29.3 ENTRY <id> SOURCE UDP PORT

Specifies a port or a range of UDP source ports. The packet must be UDP to match this criteria.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> source udp port <low-port> [<high-port>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 source udp port 10000 12000
Stateful IPV6 Access List list2>
```

2.4.3.30 ENTRY <id> STRING

Specifies a text string. The AFS system scans the packet looking for this string. If found, the packet is considered to match. By default, the comparison is not case sensitive. However, this factor will be taken into account when using the **case-sensitive** option.

Optionally, you can specify an initial search point and an end point.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> string <s> [case-sensitive] [from <fm> [to <to>]]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 string "www.sample.com"
Stateful IPV6 Access List list2>
```

2.4.3.31 ENTRY <id> TCP-FLAGS

Selects a packet based on its TCP flag values. A value (or an "OR" for them) is specified, together with a mask (Set of them). The TCP flag value adjusts to the following table:

U, URG.	0x20 Urgent pointer valid flag.
A, ACK.	0x10 Acknowledgment number valid flag.
P, PSH.	0x08 Push flag.
R, RST.	0x04 Reset connection flag.
S, SYN.	0x02 Synchronize sequence numbers flag.
F, FIN.	0x01 End of data flag.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> tcp-flags <flags> [<mask>]
```

Example:

```
Stateful IPV6 Access List list2>entry 1 tcp-flags 2 2
Stateful IPV6 Access List list2>
```

2.4.3.32 ENTRY <id> WEBSTR

Filters packets based on the content in the host or URL.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> webstr {host <hostname> | url <url>}
```

Example:

```
Stateful IPV6 Access List list2>entry 1 webstr host "hostname"
```

2.4.3.33 ENTRY <id> WEBURL

Filters packets based on content. This searches for regular text or expressions in the packets.

Syntax:

```
Stateful IPV6 Access List listName>entry <id> weburl [regex | text] <text>
```

Example:

```
Stateful IPV6 Access List list2>entry 1 weburl regex "textoquequierobuscar"
```

2.4.4 NO

Disables features, or sets the default values in some parameters.

Syntax:

```
Stateful IPv6 Access List listName>no ?
  description      Description of this rule
  entry            Configure an entry for this access list
```

2.4.4.1 NO DESCRIPTION

Deletes the textual description associated to the IPv6 Access Control List.

Syntax:

```
Stateful IPv6 Access List listName>no description
```

Example:

```
Stateful IPv6 Access List list2>no description
Stateful IPv6 Access List list2>
```

2.4.4.2 NO ENTRY

Deletes an entry from the IPv6 Access Control List. To do this, enter the identifier from the list you wish to eliminate.

Syntax:

```
Stateful IPv6 Access List listName>no entry <id>
```

Example:

```
Stateful IPv6 Access List list2>no entry 3
Stateful IPv6 Access List list2>
```

2.4.5 EXIT

Exits the IPv6 Access Control list configuration environment and returns to the main IPv6 Access Control menu prompt.

Syntax:

```
Stateful IPv6 Access List listName>exit
```

Example:

```
Stateful IPv6 Access List list2>exit
IPv6 Access Lists config>
```

2.5 Associating an access control list to the IPv6 protocol

This section focuses on how to globally associate an IPv6 access control list to the IPv6 protocol. When configured, this kind of filter only applies to local traffic (i.e. traffic that is locally generated or directed to the device). Therefore, you must decide whether to apply this filter to input (thus avoiding device overload) or to output traffic.

To associate an IPv6 access control list (configured as previously shown) to the global IPv6 protocol, enter the **protocol ipv6** command from the main configuration menu:

Syntax:

```
Config>protocol ipv6

-- IPv6 user configuration --
```

Once you are in the ipv6 protocol menu, enter the **access-group** command, followed by the name you want from the IPv6 access control list. Finally, decide whether said list is going to be applied to the input traffic (using the **in** option) or to the output traffic (using the **out** option):

Syntax:

```
IPv6 config>access-group ?
<1..50 chars>      IPv6 access-list name
IPv6 config>access-group listID ?
in      Inbound traffic filter
out     Outbound traffic filter
IPv6 config>
```

Example:

```
IPv6 config>access-group list1 in
IPv6 config>
```

The "IPv6 list1" access control list is applied to the traffic directed to the device.

2.6 Associating an IPv6 access control list to an interface

This section focuses on how to associate an IPv6 access control list to an interface.

An access control list can be associated to an interface from the interface menu. Once you are in the interface menu, enter the **ipv6 access-group** command, followed by the name you want from the IPv6 access control list. Finally, opt for alternative **in** or **out** (depending on whether you want to filter the input or output IPv6 traffic in this interface).

Syntax:

```
*config
Config>network <interface>
-- interface Interface User Configuration --
interface config>ipv6 access-group <access-list name> {in | out}
```

Example:

```
*config
Config>network ethernet0/0
-- Ethernet Interface User Configuration --
ethernet0/0 config>ipv6 access-group list1 out
```

The IPv6 access control list is applied to the traffic leaving the ethernet0/0 interface.

Chapter 3 Monitoring

3.1 Monitoring Commands

This chapter shows how to monitor the IPv6 access control lists configured in the device. The available monitoring commands are found in the monitoring menu for the **ipv6-access-list** feature:

Syntax:

```
*monitor
Console Operator
+
+feature ipv6-access-list
-- IPv6 Access Lists user console --
IPv6 Access Lists+?
  list      List IPv6 access list information
  exit      Exit to parent menu
IPv6 Access Lists+
```

3.1.1 LIST

The **list** command, followed by the **entries** option, allows you to monitor the entries for the active IPv6 access control lists configured in the router, as well as the location to which they are associated:

Syntax:

```
IPv6 Access Lists+list ?
  entries      List active IPv6 access list entries configuration
IPv6 Access Lists+
```

This command provides the following information for each IPv6 access control list:

- Type of access list: *Stateless* or *Stateful*.
- Functions where said list is applied: *Access group*, *Route-maps* or *reservation* (BRS). If this list is applied to a Route-map, it also indicates which match clause from said Route-map is being applied (*match ipv6 address* or *match ipv6 nexthop*).
- Items to which said list is applied: local traffic or a specific interface.
- Traffic affected: *input* or *output*.
- Configured entries: series of requirements and corresponding action.
- Number of IPv6 packets that must match the entry in question (*Hits*).

Example:

```
IPv6 Access Lists+list entries
Access list list1: Access group on local output
  Entry 1/permit, Hits 0, Type manual
    SRC 2001::200:200/128
    DES 2001::/16
    PROT 58-58
  Entry 2/deny, Hits 0, Type manual
    SRC ::/0
    DES ::/0
Stateful access list list2: Access group on ethernet0/0 input
  Entry 1/permit, Hits 0, Type manual
    SRC 111::111/128
    PROT ipv6-icmp
  Entry 2/permit, Hits 0, Type manual
    SRC 222::222/128
    PROT ipv6-icmp
  Entry 3/deny, Hits 0, Type manual
IPv6 Access Lists+
```

This example shows two IPv6 access control lists:

- **list1** is stateless and is applied to local output traffic. It has two entries. The first accepts traffic from the ICMPv6

protocol (protocol number 58) with source IPv6 address 2001::200:200/128 and any destination from the 2001::/16 range. The second entry rejects all other traffic.

- **list2** is stateful and is applied to the input traffic in the ethernet0/0 interface. It has three entries. The first accepts traffic with source IPv6 address 111::111/128 and ICMPv6 protocol. The second accepts traffic with IPv6 address 222::222/128 and ICMPv6 protocol. The third rejects all other traffic.

The **list entries** command allows you to filter the access lists when you want to show their entries. To do this, enter the criteria you wish to use to execute the filtering:

Syntax:

```
IPv6 Access Lists+list entries ?
  access-group      List access lists used for access groups
  access-list       List stateless access lists
  bandwidth-reservation  List access lists used for bandwidth reservation
  in               List access lists applied to input traffic
  interface        List access lists applied to an interface
  local            List access lists applied to local traffic
  out              List access lists applied to output traffic
  route-map        List access lists used for route maps
  stateful-access-list  List stateful access lists
  <cr>
```

You can display the access lists entries according to the following:

- Type of access list: using the **access-list** option to select from among the stateless lists, or the **stateful-access-list** option to select from among the statefuls, pick a single list through the name option (followed by the access list **name**). You can also select the same type of lists using the **all** option.
- The feature the access list has been associated to: with the **access-group**, **route-map** or **bandwidth-reservation** options.
- The interface where the access list is applied: with the **interface** option followed by the name of the required interface.
- If the access list is going to be applied to the local traffic (this is only logical for the access-group features): with the **local** option.
- The address of the traffic affected by the access list: with the **in** and **out** options (when the feature is BRS, this is only logical for **output** traffic.)
- If you select the **route-map** option as the feature associated to the access list, you can filter using the route-map match clause (**address** or **next-hop**):

```
IPv6 Access Lists+list entries route-map ?
  access-list       List stateless access lists
  in               List access lists applied to input traffic
  interface        List access lists applied to an interface
  local            List access lists applied to local traffic
  out              List access lists applied to output traffic
  match-address     List access lists applied in route-map address match clause
  match-next-hop    List access lists applied in route-map next-hop match clause
  <cr>
```

Example1:

```
IPv6 Access Lists+list entries access-list name list1
Access list list1: Access group on local output
  Entry 1/permit, Hits 0, Type manual
    SRC 2001::200:200/128
    DES 2001::/16
    PROT 58-58
  Entry 2/deny, Hits 6, Type manual
    (fe80::2a0:26ff:fe44:0 <-> ff02::2)  ICMPv6  TYPE=133 CODE=0  RS
    SRC ::/0
    DES ::/0
IPv6 Access Lists+
```

Example2:

```
IPv6 Access Lists+list entries access-group interface ethernet0/0 in
Stateful access list list2: Access group on ethernet0/0 input
```

```
Entry 1/permit, Hits 0, Type manual
    SRC: 111::111/128
    PROTOCOL: ipv6-icmp
Entry 2/permit, Hits 0, Type manual
    SRC: 222::222/128
    PROTOCOL: ipv6-icmp
Entry 3/deny, Hits 0, Type manual
IPv6 Access Lists+
```

Chapter 4 Configuration Examples

4.1 Scenario 1

Scenario 1 shows a router executing filtering functions between the two networks it is connected to. This scenario is shown in Figure 3.

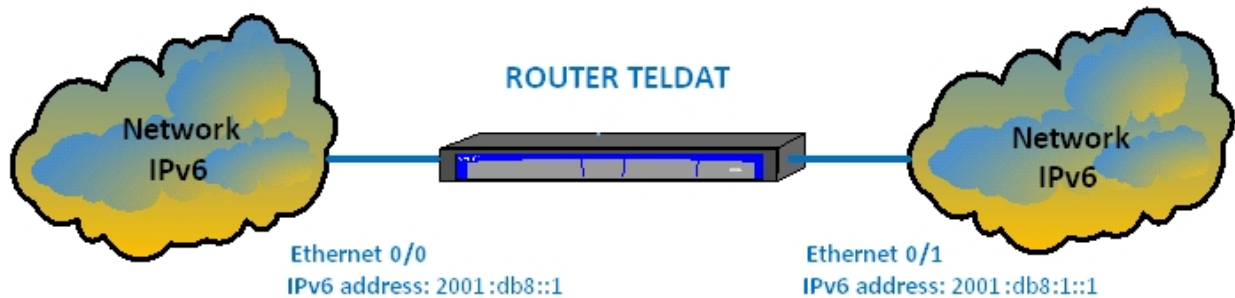


Fig. 3: Scenario 1

The router shown in Figure 3 carries out the following filtering tasks over the IPv6 packets:

- **Traffic directed towards the router**

Not compatible with Telnet network protocol.

- **Locally generated traffic**

This drops all packets with ICMPv6 protocol. The remaining traffic is permitted.

- **Traffic entering through ethernet0/1**

This only allows traffic from previously established tcp connections. The rest is dropped.

- **Traffic leaving through ethernet0/1**

This only allows packets with source network 2001:db8::/64 and destination network 2001:db8:1::/64. The remaining traffic is dropped.

- **Traffic entering through ethernet0/0**

This rejects all traffic with DSCP 0. The remaining traffic is permitted.

- **Traffic leaving through ethernet0/0**

This only allows traffic with destination network 2001:db8::/64 and any destination port belonging to the 1024-65535 range. The remaining traffic is dropped.

The final configuration for the router is as follows:

```
log-command-errors
no configuration
set data-link x25 serial0/1
feature ipv6-access-list
; -- IPv6 Access Lists user configuration --
    access-list list1
        entry 1 deny
        entry 1 protocol icmp
;
    entry 2 permit
;
exit
;
access-list list2
    entry 1 deny
    entry 1 destination port 23 23
    entry 1 protocol tcp
;
```



```

        entry 2 permit
;
    exit
;
    access-list list3
        entry 1 permit
        entry 1 tcp-specific established-state
        entry 1 protocol tcp
;
        entry 2 deny
        entry 2 protocol tcp
;
        entry 3 permit
;
    exit
;
    access-list list4
        entry 1 permit
        entry 1 source address 2001:db8::/64
        entry 1 destination address 2001:db8:1::/64
;
        entry 2 deny
;
exit
;
    access-list list5
        entry 1 deny
        entry 1 dscp 0
;
        entry 2 permit
;
    exit
;
    access-list list6
        entry 1 permit
        entry 1 destination address 2001:db8::/64
        entry 1 destination port 1024 65535
;
        entry 2 deny
;
    exit
;
    exit
;
    network ethernet0/0
; -- Ethernet Interface User Configuration -
        ipv6 enable
        ipv6 address 2001:db8::1/64
        ipv6 access-group list6 out
        ipv6 access-group list5 in
    exit
;
    network ethernet0/1
; -- Ethernet Interface User Configuration -
        ipv6 enable
        ipv6 address 2001:db8:1::1/64
        ipv6 access-group list4 out
        ipv6 access-group list3 in
    exit
;
;
    protocol ipv6
; -- IPv6 user configuration --
        access-group list1 out
        access-group list2 in
    exit
;

```

```
;
dump-command-errors
end
```

**Note**

Events 69 and 70 from the IP6 subsystem provide information on IPv6 access control.

Chapter 5 Annex

5.1 Reserved Ports

In the TCP and UDP transport layer protocols, widely used over IP version 6 (IPv6) [RFC2460], there is a field known as port. This field is made up of 16 bits.

TCP uses ports to name logical connection ends whenever a conversation is maintained. In order to provide services to unknown callers, a contact port is defined for said service. There is a list that assigns port numbers, which are pre-determined for specific services.

For possible expansions, this same port assignation is used with UDP.

The port numbers are divided into three ranges:

- Reserved (0-1023).
- Registered (1024-49151).
- Dynamic or private (49152-65535).

The following list shows some of the most used Reserved Ports:

Keyword	Decimal	Description
ftp-data	20/tcp	File Transfer [Default Data]
ftp-data	20/udp	File Transfer [Default Data]
ftp	21/tcp	File Transfer [Control]
ftp	21/udp	File Transfer [Control]
telnet	23/tcp	Telnet
telnet	23/udp	Telnet
smtp	25/tcp	Simple Mail Transfer
smtp	25/udp	Simple Mail Transfer
nameserver	42/tcp	Host Name Server
nameserver	42/udp	Host Name Server
domain	53/tcp	Domain Name Server
domain	53/udp	Domain Name Server
tftp	69/tcp	Trivial File Transfer
tftp	69/udp	Trivial File Transfer
gopher	70/tcp	Gopher
gopher	70/udp	Gopher
http	80/tcp	World Wide Web HTTP
http	80/udp	World Wide Web HTTP
snmp	161/tcp	SNMP
snmp	161/udp	SNMP
snmptrap	162/tcp	SNMPTRAP
snmptrap	162/udp	SNMPTRAP

5.2 Reserved Protocols

In IP version 6 (IPv6) [RFC2460], you'll find a field titled NextHeader. This identifies the next protocol layer.

The following Internet Protocol numbers are assigned:

Decimal	Keyword	Protocol	References
0	HOPOPT	IPv6 Hop-by-Hop Option	[RFC1883]
1	ICMP	Internet Control Message	[RFC792]
2	IGMP	Internet Group Management	[RFC1112]
3	GGP	Gateway-to-Gateway	[RFC823]

4	IP	IP in IP (encapsulation)	[RFC2003]
5	ST	Stream	[RFC1190,RFC1819]
6	TCP	Transmission Control	[RFC793]
7	CBT	CBT	[Ballardie]
8	EGP	Exterior Gateway Protocol	[RFC888,DLM1]
9	IGP	Any private interior gateway (used by Cisco for their IGRP)	[IANA]
10	BBN-RCC-MON	BBN RCC Monitoring	[SGC]
11	NVP-II	Network Voice Protocol	[RFC741,SC3]
12	PUP	PUP	[PUP,XEROX]
13	ARGUS	ARGUS	[RWS4]
14	EMCON	EMCON	[BN7]
15	XNET	Cross Net Debugger	[IEN158,JFH2]
16	CHAOS	Chaos	[NC3]
17	UDP	User Datagram	[RFC768,JBP]
18	MUX	Multiplexing	[IEN90,JBP]
19	DCN-MEAS	DCN Measurement Subsystems	[DLM1]
20	HMP	Host Monitoring	[RFC869,RH6]
21	PRM	Packet Radio Measurement	[ZSU]
22	XNS-IDP	XEROX NS IDP	[ETHERNET,XEROX]
23	TRUNK-1	Trunk-1	[BWB6]
24	TRUNK-2	Trunk-2	[BWB6]
25	LEAF-1	Leaf-1	[BWB6]
26	LEAF-2	Leaf-2	[BWB6]
27	RDP	Reliable Data Protocol	[RFC908,RH6]
28	IRTP	Internet Reliable Transaction	[RFC938,TXM]
29	ISO-TP4	ISO Transport Protocol Class 4	[RFC905,RC77]
30	NETBLT	Bulk Data Transfer Protocol	[RFC969,DDC1]
31	MFE-NSP	MFE Network Services Protocol	[MFENET,BCH2]
32	MERIT-INP	MERIT Internodal Protocol	[HWB]
33	SEP	Sequential Exchange Protocol	[JC120]
34	3PC	Third Party Connect Protocol	[SAF3]
35	IDPR	Inter-Domain Policy Routing Protocol	[MXS1]
36	XTP	XTP	[GXC]
37	DDP	Datagram Delivery Protocol	[WXC]
38	IDPR-CMTP	IDPR Control Message Transport Proto	[MXS1]
39	TP++	TP++ Transport Protocol	[DXF]
40	IL	IL Transport Protocol	[Presotto]
41	IPv6	Ipv6	[Deering]
42	SDRP	Source Demand Routing Protocol	[DXE1]
43	IPv6-Route	Routing Header for IPv6	[Deering]
44	IPv6-Frag	Fragment Header for IPv6	[Deering]
45	IDRP	Inter-Domain Routing Protocol	[Sue Hares]
46	RSVP	Reservation Protocol	[Bob Braden]
47	GRE	General Routing Encapsulation	[Tony Li]
48	MHRP	Mobile Host Routing Protocol	[David Johnson]
49	BNA	BNA	[Gary Salamon]
50	ESP	Encapsulating Security Payload	[RFC1827]
51	AH	Authentication Header	[RFC1826]
52	I-NLSP	Integrated Net Layer Security TUBA	[GLENN]
53	SWIPE	IP with Encryption	[JI6]
54	NARP	NBMA Address Resolution Protocol	[RFC1735]

55	MOBILE	IP Mobility	[Perkins]
56	TLSP	Transport Layer Security Protocol using Kryptonet key management	[Oberg]
57	SKIP	SKIP	[Markson]
58	IPv6-ICMP	ICMP for IPv6	[RFC1883]
59	IPv6-NoNxt	No Next Header for IPv6	[RFC1883]
60	IPv6-Opts	Destination Options for IPv6	[RFC1883]
61		any host internal protocol	[IANA]
62	CFTP	CFTP	[CFTP,HCF2]
63		Any local network	[IANA]
64	SAT-EXPAK	SATNET and Backroom EXPAK	[SHB]
65	KRYPTOLAN	Kryptolan	[PXL1]
66	RVD	MIT Remote Virtual Disk Protocol	[MBG]
67	IPPC	Internet Pluribus Packet Core	[SHB]
68		Any distributed file system	[IANA]
69	SAT-MON	SATNET Monitoring	[SHB]
70	VISA	VISA Protocol	[GXT1]
71	IPCV	Internet Packet Core Utility	[SHB]
72	CPNX	Computer Protocol Network Executive	[DXM2]
73	CPHB	Computer Protocol Heart Beat	[DXM2]
74	WSN	Wang Span Network	[VXD]
75	PVP	Packet Video Protocol	[SC3]
76	BR-SAT-MON	Backroom SATNET Monitoring	[SHB]
77	SUN-ND	SUN ND PROTOCOL-Temporary	[WM3]
78	WB-MON	WIDEBAND Monitoring	[SHB]
79	WB-EXPAK	WIDEBAND EXPAK	[SHB]
80	ISO-IP	ISO Internet Protocol	[MTR]
81	VMTP	VMTP	[DRC3]
82	SECURE-VMTP	SECURE-VMTP	[DRC3]
83	VINES	VINES	[BXH]
84	TTP	TTP	[JXS]
85	NSFNET-IGP	NSFNET-IGP	[HWB]
86	DGP	Dissimilar Gateway Protocol	[DGP,ML109]
87	TCF	TCF	[GAL5]
88	EIGRP	EIGRP	[CISCO,GXS]
89	OSPFGRP	OSPFGRP	[RFC1583,JTM4]
90	Sprite-RPC	Sprite RPC Protocol	[SPRITE,BXW]
91	LARP	Locus Address Resolution Protocol	[BXH]
92	MTP	Multicast Transport Protocol	[SXA]
93	AX.25	AX.25 Frames	[BK29]
94	IPIP	IP-within-IP Encapsulation Protocol	[JI6]
95	MICP	Mobile Internetworking Control Pro.	[JI6]
96	SCC-SP	Semaphore Communications Sec. Pro.	[HXH]
97	ETHERIP	Ethernet-within-IP Encapsulation	[RDH1]
98	ENCAP	Encapsulation Header	[RFC1241,RXB3]
99		Any private encryption scheme	[IANA]
100	GMTP	GMTP	[RXB5]
101	IFMP	Ipsilon Flow Management Protocol	[Hinden]
102	PNNI	PNNI over IP	[Callon]
103	PIM	Protocol Independent Multicast	[Farinacci]
104	ARIS	ARIS	[Feldman]
105	SCPS	SCPS	[Durst]

106	QNX	QNX	[Hunter]
107	A/N	Active Networks	[Braden]
108	IPComp	IP Payload Compression Protocol	[RFC2393]
109	SNP	Sitara Networks Protocol	[Sridhar]
110	Compaq-Peer	Compaq Peer Protocol	[Volpe]
111	IPX-in-IP	IPX in IP	[Lee]
112	VRRP	Virtual Router Redundancy Protocol	[Hinden]
113	PGM	PGM Reliable Transport Protocol	[Speakman]
114		Any 0-hop protocol	[IANA]
115	L2TP	Layer Two Tunneling Protocol	[Aboba]
116	DDX	D-II Data Exchange (DDX)	[Worley]
117	IATP	Interactive Agent Transfer Protocol	[Murphy]
118	STP	Schedule Transfer Protocol	[JMP]
119	SRP	SpectraLink Radio Protocol	[Hamilton]
120	UTI	UTI	[Lothberg]
121	SMP	Simple Message Protocol	[Ekblad]
122	SM	SM	[Crowcroft]
123	PTP	Performance Transparency Protocol	[Welzl]
124	ISIS over IPv4		[Przygienda]
125	FIRE		[Partridge]
126	CRTP	Combat Radio Transport Protocol	[Sautter]
127	CRUDP	Combat Radio User Datagram	[Sautter]
128	SSCOPMCE		[Waber]
129	IPLT		[Hollbach]
130	SPS	Secure Packet Shield	[McIntosh]
131	PIPE	Private IP Encapsulation within IP	[Petri]
132	SCTP	Stream Control Transmission Protocol	[Stewart]
133	FC	FibreChannel	[Rajagopal]
134	RSVP-E2E-IGNORE		[RFC3175]
135-254		Unassigned	[IANA]
255		Reserved	[IANA]