[[Search](https://datatracker.ietf.org/)] [[txt](https://www.ietf.org/archive/id/draft-dbb-netmod-acl-01.txt" \o "Plaintext version of this document)|[pdfized](https://datatracker.ietf.org/doc/pdf/draft-dbb-netmod-acl-01)|[bibtex](https://datatracker.ietf.org/doc/draft-dbb-netmod-acl/01/bibtex/)] [[Tracker](https://datatracker.ietf.org/doc/draft-dbb-netmod-acl/)] [[Email](mailto:draft-dbb-netmod-acl@ietf.org?subject=draft-dbb-netmod-acl)] [[Diff1](https://www.ietf.org/rfcdiff?difftype=--hwdiff&url2=draft-dbb-netmod-acl-01.txt)] [[Diff2](https://www.ietf.org/rfcdiff?url2=draft-dbb-netmod-acl-01.txt)] [[Nits](https://author-tools.ietf.org/api/idnits?url=https://www.ietf.org/archive/id/draft-dbb-netmod-acl-01.txt)]

Versions: [00](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl-00) [**01**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl-01)

netmod O. G. D. Dios

Internet-Draft S. Barguil

Intended status: Standards Track Telefonica

Expires: 31 December 2022 M. Boucadair

Orange

29 June 2022

**Extensions to the Access Control Lists (ACLs) YANG Model**

**draft-dbb-netmod-acl-01**

Abstract

[RFC 8519](https://datatracker.ietf.org/doc/html/rfc8519) defines a YANG data model for Access Control Lists (ACLs).

This document discusses a set of extensions that fix many of the

limitations of the ACL model as initially defined in [RFC 8519](https://datatracker.ietf.org/doc/html/rfc8519).

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the Network Modeling

Working Group mailing list (netmod@ietf.org), which is archived at

<https://mailarchive.ietf.org/arch/browse/netmod/>.

Source for this draft and an issue tracker can be found at

<https://github.com/oscargdd/draft-dbb-netmod-enhanced-acl>.

Status of This Memo

This Internet-Draft is submitted in full conformance with the

provisions of [BCP 78](https://datatracker.ietf.org/doc/html/bcp78) and [BCP 79](https://datatracker.ietf.org/doc/html/bcp79).

Internet-Drafts are working documents of the Internet Engineering

Task Force (IETF). Note that other groups may also distribute

working documents as Internet-Drafts. The list of current Internet-

Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months

and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference

material or to cite them other than as "work in progress."

This Internet-Draft will expire on 31 December 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the

document authors. All rights reserved.

This document is subject to [BCP 78](https://datatracker.ietf.org/doc/html/bcp78) and the IETF Trust's Legal

Provisions Relating to IETF Documents ([https://trustee.ietf.org/](https://trustee.ietf.org/license-info)

[license-info](https://trustee.ietf.org/license-info)) in effect on the date of publication of this document.

Please review these documents carefully, as they describe your rights

and restrictions with respect to this document. Code Components

extracted from this document must include Revised BSD License text as

described in Section 4.e of the [Trust Legal Provisions](https://trustee.ietf.org/license-info) and are

provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

2. Terminology

3. Problem Statement & Gap Analysis

3.1. Suboptimal Configuration: Lack of Manipulating Lists of

Prefixes

3.2. Manageability: Impossibility to Use Aliases or Defined Sets

3.3. Bind ACLs to Devices, Not Only Interfaces

3.4. Partial or Lack of IPv4/IPv6 Fragment Handling

3.5. Suboptimal TCP Flags Handling

3.6. Rate-Limit Action

3.7. Payload-based Filtering

3.8. Reuse the ACLs Content Across Several Devices

4. Overall Module Structure

4.1. Enhanced ACL

4.2. Defined sets

4.3. TCP Flags Handling

4.4. Fragments Handling

4.5. Rate-Limit Traffic

5. YANG Modules

5.1. Enhanced ACL

6. Security Considerations (TBC)

7. IANA Considerations

7.1. URI Registration

7.2. YANG Module Name Registration

8. References

8.1. Normative References

8.2. Informative References

[Appendix A](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#appendix-A). Acknowledgements

Authors' Addresses

[**1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-1)**. Introduction**

[RFC8519] defines Access control lists (ACLs) as a user-ordered set of

filtering rules. The model targets the configuration of the

filtering behaviour of a device. However, the model structure, as

defined in [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)], suffers from a set of limitations. This

document describes these limitations and proposes an enhanced ACL

structure. The YANG module in this document is solely based on

augmentations to the ACL YANG module defined in [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)].

The motivation of such enhanced ACL structure is discussed in detail

in [Section 3](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3).

When managing ACLs, it is common for network operators to group

match elements in pre-defined sets. The consolidation into

group matches allows for reducing the number of rules, especially in large

scale networks. If it is needed, for example, to find a match

against 100 IP addresses (or prefixes), a single rule will suffice

rather than creating individual Access Control Entries (ACEs) for

each IP address (or prefix). In doing so, implementations would

optimize the performance of matching lists vs multiple rules

matching.

The enhanced ACL structure is also meant to facilitate the management

of network operators. Instead of entering the IP address or port

number literals, using user-named lists decouples the creation of the

rule from the management of the sets. Hence, it is possible to

remove/add entries to the list without redefining the (parent) ACL

rule.

In addition, the notion of Access Control List (ACL) and defined sets

is generalized so that it is not device-specific as per [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)].

ACLs and defined sets may be defined at network / administrative

domain level and associated to devices. This approach facilitates

the reusability across multiple network elements. For example,

managing the IP prefix sets from a network level makes it easier to

maintain by the security groups.

Network operators maintain sets of IP prefixes that are related to

each other, e.g., deny-lists or accept-lists that are associated with

those provided by a VPN customer. These lists are maintained and

manipulated by security expert teams.

Note that ACLs are used locally in devices but are triggered by other

tools such as DDoS mitigation [[RFC9132](https://datatracker.ietf.org/doc/html/rfc9132)] or BGP Flow Spec [[RFC8955](https://datatracker.ietf.org/doc/html/rfc8955)]

[[RFC8956](https://datatracker.ietf.org/doc/html/rfc8956)]. Therefore, supporting means to easily map to the

filtering rules conveyed in messages triggered by these tools is

valuable from a network operation standpoint.

[**2**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-2)**. Terminology**

The keywords \*MUST\*, \*MUST NOT\*, \*REQUIRED\*, \*SHALL\*, \*SHALL NOT\*,

\*SHOULD\*, \*SHOULD NOT\*, \*RECOMMENDED\*, \*MAY\*, and \*OPTIONAL\*, when

they appear in this document, are to be interpreted as described in

[[RFC2119](https://datatracker.ietf.org/doc/html/rfc2119)].

The terminology for describing YANG modules is defined in [[RFC7950](https://datatracker.ietf.org/doc/html/rfc7950)].

The meaning of the symbols in the tree diagrams is defined in

[[RFC8340](https://datatracker.ietf.org/doc/html/rfc8340)].

In addition to the terms defined in [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)], this document makes use

of the following terms:

\* Defined set: Refers to reusable description of one or multiple

information elements (e.g., IP address, IP prefix, port number,

or ICMP type).

[**3**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3)**. Problem Statement & Gap Analysis**

[**3.1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.1)**. Suboptimal Configuration: Lack of Support for Lists of Prefixes**

IP prefix related data nodes, e.g., "destination-ipv4-network" or

"destination-ipv6-network", do not support handling a list of IP

prefixes, which may then lead to having to support large numbers of ACL entries in a configuration file. The same issue

is encountered when ACLs have to be in place to mitigate DDoS attacks

(e.g., [[RFC9132](https://datatracker.ietf.org/doc/html/rfc9132)] when a set of sources are involved in such an

attack. The situation is even worse when both a list of sources and

destination prefixes are involved.

Figure 1 shows an example of the required ACL configuration for

filtering traffic from two prefixes.

{

"ietf-access-control-list:acls": {

"acl": [

{

"name": "first-prefix",

"type": "ipv6-acl-type",

"aces": {

"ace": [

{

"name": "my-test-ace",

"matches": {

"ipv6": {

"destination-ipv6-network":

"2001:db8:6401:1::/64",

"source-ipv6-network":

"2001:db8:1234::/96",

"protocol": 17,

"flow-label": 10000

},

"udp": {

"source-port": {

"operator": "lte",

"port": 80

},

"destination-port": {

"operator": "neq",

"port": 1010

}

}

},

"actions": {

"forwarding": "accept"

}

}

]

}

},

{

"name": "second-prefix",

"type": "ipv6-acl-type",

"aces": {

"ace": [

{

"name": "my-test-ace",

"matches": {

"ipv6": {

"destination-ipv6-network":

"2001:db8:6401:c::/64",

"source-ipv6-network":

"2001:db8:1234::/96",

"protocol": 17,

"flow-label": 10000

},

"udp": {

"source-port": {

"operator": "lte",

"port": 80

},

"destination-port": {

"operator": "neq",

"port": 1010

}

}

},

"actions": {

"forwarding": "accept"

}

}

]

}

}

]

}

}

Figure 1: Example Illustrating Sub-optimal Use of the ACL Model

with a Prefix List

Such configuration is suboptimal for both: - Network controllers that

need to manipulate large files. All or a subset for this

configuration will need to be passed to the underlying network

devices. - Devices may receive such a configuration and thus will need

to maintain it locally.

(Figure 2 depicts an example of an optimized structure:

{

"ietf-access-control-list:acls": {

"acl": [

{

"name": "prefix-list-support",

"type": "ipv6-acl-type",

"aces": {

"ace": [

{

"name": "my-test-ace",

"matches": {

"ipv6": {

"destination-ipv6-network": [

"2001:db8:6401:1::/64",

"2001:db8:6401:c::/64"

],

"source-ipv6-network":

"2001:db8:1234::/96",

"protocol": 17,

"flow-label": 10000

},

"udp": {

"source-port": {

"operator": "lte",

"port": 80

},

"destination-port": {

"operator": "neq",

"port": 1010

}

}

},

"actions": {

"forwarding": "accept"

}

}

]

}

}

]

}

}

Figure 2: Example Illustrating Optimal Use of the ACL Model in a

Network Context.

[**3.2**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.2)**. Manageability: Impossibility to Use Aliases or Defined Sets**

The same approach as the one discussed for IP prefixes can be

generalized by introducing the concept of "aliases" or "defined sets".

The defined sets are reusable definitions across several ACLs. Each

category is modelled in YANG as a list of parameters related to the

class it represents. The following sets can be considered:

\* Prefix sets: Used to create lists of IPv4 or IPv6 prefixes.

\* Protocol sets: Used to create a list of protocols.

\* Port number sets: Used to create lists of TCP or UDP port values

(or any other transport protocol that makes uses of port numbers).

The identity of the protocols is identified by the protocol set, if

present. Otherwise, a set applies to any protocol.

\* ICMP sets: Uses to create lists of ICMP-based filters. This

applies only when the protocol is set to ICMP or ICMPv6.

A candidate structure is shown in Figure 3:

+--rw defined-sets

| +--rw prefix-sets

| | +--rw prefix-set\* [name mode]

| | +--rw name string

| | +--rw ip-prefix\* inet:ip-prefix

| +--rw port-sets

| | +--rw port-set\* [name]

| | +--rw name string

| | +--rw port\* inet:port-number

| +--rw protocol-sets

| | +--rw protocol-set\* [name]

| | +--rw name string

| | +--rw protocol-name\* identityref

| +--rw icmp-type-sets

| +--rw icmp-type-set\* [name]

| +--rw name string

| +--rw types\* [type]

| +--rw type uint8

| +--rw code? uint8

| +--rw rest-of-header? binary

Figure 3: Examples of Defined Sets.

[**3.3**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.3)**. Bind ACLs to Devices, Not Only Interfaces**

In the context of network management, an ACL may be enforced in many

network locations. As such, the ACL module should allow for binding an

ACL to multiple devices, not only (abstract) interfaces.

The ACL name must, thus, be unique at the scale of the network, but

the same name may be used in many devices when enforcing node-

specific ACLs.

[**3.4**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.4)**. Partial or Lack of IPv4/IPv6 Fragment Handling**

[RFC8519] does not support fragment handling capability for IPv6 but

offers a partial support for IPv4 by means of 'flags'. Nevertheless,

the use of 'flags' is problematic since it does not allow a bitmask

to be defined. For example, setting other bits not covered by the

'flags' filtering clause in a packet will allow that packet to get

through (because it won't match the ACE).

Defining a new IPv4/IPv6 matching field called 'fragment' is thus

required to efficiently handle fragment-related filtering rules.

[**3.5**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.5)**. Suboptimal TCP Flags Handling**

[RFC8519] allows including flags in the TCP match fields, however

that structure does not support matching operations as those

supported in BGP Flow Spec. Defining this field to be defined as a

flag bitmask together with a set of operations is meant to

efficiently handle TCP flags filtering rules.

[**3.6**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.6)**. Rate-Limit Action**

[RFC8519] specifies that forwarding actions can be 'accept' (i.e.,

accept matching traffic), 'drop' (i.e., drop matching traffic without

sending any ICMP error message), or 'reject' (i.e., drop matching

traffic and send an ICMP error message to the source). However,

there are situations where the matching traffic can be accepted, but

with a rate-limit policy. Such capability is not currently supported

by [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)].

[**3.7**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.7)**. Payload-based Filtering**

Some transport protocols use existing protocols (e.g., TCP or UDP) as

substrate. The match criteria for such protocols may rely upon the

'protocol' under 'l3', TCP/UDP match criteria, part of the TCP/UDP

Payload, or a combination thereof. [[RFC8519](https://datatracker.ietf.org/doc/html/rfc8519)] does not support

matching based on the payload.

Likewise, the current version of the ACL model does not support

filtering of encapsulated traffic.

[**3.8**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.8)**. Reuse the ACLs Content Across Several Devices**

Having a global network view of the ACLs is highly valuable for

service providers. An ACL could be defined and applied following the

hierarchy of the network topology. So, an ACL can be defined at the

network level and, then, that same ACL can be used (or referenced to)

in several devices (including termination points) within the same

network.

This network/device ACLs differentiation introduces several new

requirements, e.g.:

\* An ACL name can be used at both network and device levels.

\* An ACL content updated at the network level should imply a

transaction that updates the relevant content in all the nodes

using this ACL.

\* ACLs defined at the device level have a local meaning for the

specific node.

\* A device can be associated with a router, a VRF, a logical system,

or a virtual node. ACLs can be applied in physical and logical

infrastructure.

[**4**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4)**. Overall Module Structure**

[**4.1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.1)**. Enhanced ACL**

module: ietf-acl-enh

augment /ietf-acl:acls/ietf-acl:acl:

+--rw defined-sets

+--rw ipv4-prefix-sets

| +--rw prefix-set\* [name]

| +--rw name string

| +--rw description? string

| +--rw prefix\* inet:ipv4-prefix

+--rw ipv6-prefix-sets

| +--rw prefix-set\* [name]

| +--rw name string

| +--rw description? string

| +--rw prefix\* inet:ipv6-prefix

+--rw port-sets

| +--rw port-set\* [name]

| +--rw name string

| +--rw port\* [id]

| +--rw id string

| +--rw (port)?

| +--:(port-range-or-operator)

| +--rw port-range-or-operator

| +--rw (port-range-or-operator)?

| +--:(range)

| | +--rw lower-port inet:port-number

| | +--rw upper-port inet:port-number

| +--:(operator)

| +--rw operator? operator

| +--rw port inet:port-number

+--rw protocol-sets

| +--rw protocol-set\* [name]

| +--rw name string

| +--rw protocol\* union

+--rw icmp-type-sets

+--rw icmp-type-set\* [name]

+--rw name string

+--rw types\* [type]

+--rw type uint8

+--rw code? uint8

+--rw rest-of-header? binary

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches:

+--rw (payload)?

+--:(prefix-pattern)

+--rw prefix-pattern {match-on-payload}?

+--rw offset? identityref

+--rw offset-end? uint64

+--rw operator? operator

+--rw prefix? binary

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches/ietf-acl:l3/ietf-acl:ipv4:

+--rw ipv4-fragment

| +--rw operator? operator

| +--rw type? fragment-type

+--rw source-ipv4-prefix-list? leafref

+--rw destination-ipv4-prefix-list? leafref

+--rw next-header-set? leafref

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches/ietf-acl:l3/ietf-acl:ipv6:

+--rw ipv6-fragment

| +--rw operator? operator

| +--rw type? fragment-type

+--rw source-ipv6-prefix-list? leafref

+--rw destination-ipv6-prefix-list? leafref

+--rw protocol-set? leafref

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches/ietf-acl:l4/ietf-acl:tcp:

+--rw flags-bitmask

| +--rw operator? operator

| +--rw bitmask? uint16

+--rw source-tcp-port-set?

| -> ../../../../defined-sets/port-sets/port-set/name

+--rw destination-tcp-port-set?

-> ../../../../defined-sets/port-sets/port-set/name

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches/ietf-acl:l4/ietf-acl:udp:

+--rw source-udp-port-set?

| -> ../../../../defined-sets/port-sets/port-set/name

+--rw destination-udp-port-set?

-> ../../../../defined-sets/port-sets/port-set/name

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:matches/ietf-acl:l4/ietf-acl:icmp:

+--rw icmp-set? leafref

augment /ietf-acl:acls/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace

/ietf-acl:actions:

+--rw rate-limit? decimal64

Figure 4: Enhanced ACL tree

[**4.2**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.2)**. Defined sets**

The augmented ACL structure includes several containers to manage

reusable sets of elements that can be matched in an ACL entry. Each

set is uniquely identified by a name, and can be called from the

relevant entry. The following sets are defined:

\* IPv4 Prefix set: It contains a list of IPv4 prefixes. A match

will be considered if the IP address (source or destination,

depending on the ACL entry) is contained in any of the prefixes.

\* IPv6 Prefix set: It contains a list of IPv6 prefixes. A match

will be considered if the IP address (source or destination,

depending on the ACL entry) is contained in any of the prefixes.

\* Port sets: It contains a list of port numbers to be used in TCP /

UDP entries. The ports can be individual port numbers, a range of

ports, and an operation.

\* Protocol sets: It contains a list of protocol values. Each

protocol can be identified either by a number (e.g., 17) or a name

(e.g., UDP).

\* ICMP sets: It contains a list of ICMP types, each of them

identified by a type value, optionally the code and the rest of

the header.

[**4.3**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.3)**. TCP Flags Handling**

The augmented ACL structure includes a new leaf 'flags-bitmask' to

better handle flags.

Clients that support both 'flags-bitmask' and 'flags' matching fields

MUST NOT set these fields in the same request.

Figure 5 shows an example of a request to install a filter to discard

incoming TCP messages having all flags unset.

{

"ietf-access-control-list:acls": {

"acl": [{

"name": "tcp-flags-example",

"aces": {

"ace": [{

"name": "null-attack",

"matches": {

"tcp": {

"flags-bitmask": {

"operator": "not any",

"bitmask": 4095

}

}

},

"actions": {

"forwarding": "drop"

}

}]

}

}]

}

}

Figure 5: Example to Deny TCP Null Attack Messages

[**4.4**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.4)**. Fragments Handling**

The augmented ACL structure includes a new leaf 'fragment' to better

handle fragments.

Clients that support both 'fragment' and 'flags' matching fields MUST

NOT set these fields in the same request.

Figure 6 shows the content of a POST request to allow the traffic

destined to 198.51.100.0/24 and UDP port number 53, but to drop all

fragmented packets. The following ACEs are defined (in this order):

\* "drop-all-fragments" ACE: discards all fragments.

\* "allow-dns-packets" ACE: accepts DNS packets destined to

198.51.100.0/24.

{

"ietf-access-control-list:acls": {

"acl": [

{

"name": "dns-fragments",

"type": "ipv4-acl-type",

"aces": {

"ace": [

{

"name": "drop-all-fragments",

"matches": {

"ipv4": {

"ipv4-fragment": {

"operator": "match",

"type": "isf"

}

}

},

"actions": {

"forwarding": "drop"

}

},

{

"name": "allow-dns-packets",

"matches": {

"ipv4": {

"destination-ipv4-network": "198.51.100.0/24"

},

"udp": {

"destination-port": {

"operator": "eq",

"port": 53

}

},

"actions": {

"forwarding": "accept"

}

}

}

]

}

}

]

}

}

Figure 6: Example Illustrating Candidate Filtering of IPv4

Fragmented Packets.

Figure 7 shows an example of the body of a POST request to allow the

traffic destined to 2001:db8::/32 and UDP port number 53, but to drop

all fragmented packets. The following ACEs are defined (in this

order):

\* "drop-all-fragments" ACE: discards all fragments (including atomic

fragments). That is, IPv6 packets that include a Fragment header

(44) are dropped.

\* "allow-dns-packets" ACE: accepts DNS packets destined to

2001:db8::/32.

{

"ietf-access-control-list:acls": {

"acl": [

{

"name": "dns-fragments",

"type": "ipv6-acl-type",

"aces": {

"ace": [

{

"name": "drop-all-fragments",

"matches": {

"ipv6": {

"ipv6-fragment": {

"operator": "match",

"type": "isf"

}

}

},

"actions": {

"forwarding": "drop"

}

},

{

"name": "allow-dns-packets",

"matches": {

"ipv6": {

"destination-ipv6-network": "2001:db8::/32"

},

"udp": {

"destination-port": {

"operator": "eq",

"port": 53

}

}

},

"actions": {

"forwarding": "accept"

}

}

]

}

}

]

}

}

Figure 7: Example Illustrating Candidate Filtering of IPv6

Fragmented Packets.

[**4.5**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.5)**. Rate-Limit Traffic**

In order to support rate-limiting (see [Section 3.6](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-3.6)), a new action

called "rate-limit" is defined.

(#example\_5) shows an ACL example to rate-limit incoming SYNs during

a SYN flood attack.

{

"ietf-access-control-list:acls": {

"acl": [{

"name": "tcp-flags-example-with-rate-limit",

"aces": {

"ace": [{

"name": "rate-limit-syn",

"matches": {

"tcp": {

"flags-bitmask": {

"operator": "match",

"bitmask": 2

}

}

},

"actions": {

"forwarding": "accept",

"rate-limit": "20.00"

}

}]

}

}]

}

}

Figure 8: Example Rate-Limit Incoming TCP SYNs

[**5**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-5)**. YANG Modules**

[**5.1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-5.1)**. Enhanced ACL**

<CODE BEGINS> file "ietf-acl-enh@2022-06-16.yang"

module ietf-acl-enh {

yang-version 1.1;

namespace "urn:ietf:params:xml:ns:yang:ietf-acl-enh";

prefix enh-acl;

import ietf-inet-types {

prefix inet;

reference

"[RFC 6991](https://datatracker.ietf.org/doc/html/rfc6991): Common YANG Data Types";

}

import ietf-access-control-list {

prefix ietf-acl;

reference

"[RFC 8519](https://datatracker.ietf.org/doc/html/rfc8519): YANG Data Model for Network Access

Control Lists (ACLs), [Section 4.1](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.1)";

}

import ietf-packet-fields {

prefix packet-fields;

reference

"[RFC 8519](https://datatracker.ietf.org/doc/html/rfc8519): YANG Data Model for Network Access

Control Lists (ACLs), [Section 4.2](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4.2)";

}

organization

"IETF NETMOD Working Group";

contact

"WG Web: <<https://datatracker.ietf.org/wg/netmod/>>

WG List: <mailto:netmod@ietf.org>

Author: Mohamed Boucadair

<mailto:mohamed.boucadair@orange.com>

Author: Samier Barguil

<mailto:samier.barguilgiraldo.ext@telefonica.com>

Author: Oscar Gonzalez de Dios

<mailto:oscar.gonzalezdedios@telefonica.com>";

description

"This module contains YANG definitions for enhanced ACLs.

Copyright (c) 2022 IETF Trust and the persons identified as

authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or

without modification, is permitted pursuant to, and subject

to the license terms contained in, the Revised BSD License

set forth in [Section 4](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-4).c of the IETF Trust's Legal Provisions

Relating to IETF Documents

([http://trustee.ietf.org/license-info](https://trustee.ietf.org/license-info)).

This version of this YANG module is part of RFC XXXX; see

the RFC itself for full legal notices.";

revision 2022-06-16 {

description

"Initial revision.";

reference

"RFC XXXX: xxxxx";

}

feature match-on-payload {

description

"Match based on a pattern is supported.";

}

identity offset-type {

description

"Base identity for payload offset type.";

}

identity layer3 {

base offset-type;

description

"IP header.";

}

identity layer4 {

base offset-type;

description

"Transport header (e.g., TCP or UDP).";

}

identity payload {

base offset-type;

description

"Transport payload. For example, this represents the beginning

of the TCP data right after any TCP options.";

}

typedef operator {

type bits {

bit not {

position 0;

description

"If set, logical negation of operation.";

}

bit match {

position 1;

description

"Match bit. If set, this is a bitwise match operation

defined as '(data & value) == value'; if unset, (data &

value) evaluates to TRUE if any of the bits in the value

mask are set in the data , i.e., '(data & value) != 0'.";

}

}

description

"How to apply the defined bitmask.";

}

typedef fragment-type {

type bits {

bit df {

position 0;

description

"Don't fragment bit for IPv4.

Must be set to 0 when it appears in an IPv6 filter.";

}

bit isf {

position 1;

description

"Is a fragment.";

}

bit ff {

position 2;

description

"First fragment.";

}

bit lf {

position 3;

description

"Last fragment.";

}

}

description

"Different fragment types to match against.";

}

grouping tcp-flags {

description

"Operations on TCP flags.";

leaf operator {

type operator;

default "match";

description

"How to interpret the TCP flags.";

}

leaf bitmask {

type uint16;

description

"Bitmask values can be encoded as a 1- or 2-byte bitmask.

When a single byte is specified, it matches byte 13

of the TCP header, which contains bits 8 though 15

of the 4th 32-bit word. When a 2-byte encoding is used,

it matches bytes 12 and 13 of the TCP header with

the bitmask fields corresponding to the TCP data offset

field being ignored for purposes of matching.";

}

}

grouping fragment-fields {

description

"Operations on fragment types.";

leaf operator {

type operator;

default "match";

description

"How to interpret the fragment type.";

}

leaf type {

type fragment-type;

description

"What fragment type to look for.";

}

}

grouping payload {

description

"Operations on payload match.";

leaf offset {

type identityref {

base offset-type;

}

description

"Indicates the payload offset.";

}

leaf offset-end {

type uint64;

description

"Indicates the number of bytes to cover when

performing the prefix match.";

}

leaf operator {

type operator;

default "match";

description

"How to interpret the prefix match.";

}

leaf prefix {

type binary;

description

"The pattern to match against.";

}

}

augment "/ietf-acl:acls/ietf-acl:acl" {

description

"add a new container to store sets (prefix

sets, port sets, etc";

container defined-sets {

description

"Predefined sets of attributes used in policy match

statements.";

container ipv4-prefix-sets {

description

"Data definitions for a list of IPv4 or IPv6

prefixes which are matched as part of a policy.";

list prefix-set {

key "name";

description

"List of the defined prefix sets";

leaf name {

type string;

description

"Name of the prefix set -- this is used as a label to

reference the set in match conditions.";

}

leaf description {

type string;

description

"Defined Set description";

}

leaf-list prefix {

type inet:ipv4-prefix;

description

"List of IPv4 prefixes to be used in match

conditions.";

}

}

}

container ipv6-prefix-sets {

description

"Data definitions for a list of IPv6 prefixes

which are matched as part of a policy.";

list prefix-set {

key "name";

description

"List of the defined prefix sets";

leaf name {

type string;

description

"Name of the prefix set -- this is used as a label to

reference the set in match conditions.";

}

leaf description {

type string;

description

"A textual description of the prefix list.";

}

leaf-list prefix {

type inet:ipv6-prefix;

description

"List of IPv6 prefixes to be used in match

conditions.";

}

}

}

container port-sets {

description

"Data definitions for a list of ports which can

be matched in policies.";

list port-set {

key "name";

description

"List of port set definitions.";

leaf name {

type string;

description

"Name of the portset -- this is used as a label to

reference the set in match conditions.";

}

list port {

key "id";

description

"Port numbers along with the operator on which to

match.";

leaf id {

type string;

description

"Identifier of the list of ports.";

}

choice port {

description

"Choice of specifying the port number or referring

to a group of port numbers.";

container port-range-or-operator {

description

"Indicates a set of ports.";

uses packet-fields:port-range-or-operator;

}

}

}

}

}

container protocol-sets {

description

"Data definitions for a list of protocols which can

be matched in policies.";

list protocol-set {

key "name";

description

"List of protocol set definitions.";

leaf name {

type string;

description

"Name of the protocols set -- this is used as a label to

reference the set in match conditions.";

}

leaf-list protocol {

type union {

type uint8;

type string; //Check if we can reuse an IANA-maintained module

}

description

"Value of the protocol set.";

}

}

}

container icmp-type-sets {

description

"Data definitions for a list of ICMP types which can

be matched in policies.";

list icmp-type-set {

key "name";

description

"List of ICMP type set definitions.";

leaf name {

type string;

description

"Name of the ICMP type set -- this is used as a label to

reference the set in match conditions.";

}

list types {

key "type";

description

"Includes a list of ICMP types.";

uses packet-fields:acl-icmp-header-fields;

}

}

}

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches" {

description

"Add a new match types.";

choice payload {

description

"Match a prefix pattern.";

container prefix-pattern {

if-feature "match-on-payload";

description

"Rule to perform payload-based match.";

uses payload;

}

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches/ietf-acl:l3/ietf-acl:ipv4" {

description

"Handle non-initial and initial fragments for IPv4 packets.";

container ipv4-fragment {

description

"Indicates how to handle IPv4 fragments.";

uses fragment-fields;

}

leaf source-ipv4-prefix-list {

type leafref {

path "../../../../defined-sets/ipv4-prefix-sets/prefix-set/name";

}

description

"reference to a prefix list to match the source address";

}

leaf destination-ipv4-prefix-list {

type leafref {

path "../../../../defined-sets/ipv4-prefix-sets/prefix-set/name";

}

description

"reference to a prefix list to match the destination address";

}

leaf next-header-set {

type leafref {

path "../../../../defined-sets/protocol-sets/protocol-set/name";

}

description

"reference to a protocol set to match the next-header field";

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches/ietf-acl:l3/ietf-acl:ipv6" {

description

"Handle non-initial and initial fragments for IPv6 packets.";

container ipv6-fragment {

description

"Indicates how to handle IPv6 fragments.";

uses fragment-fields;

}

leaf source-ipv6-prefix-list {

type leafref {

path "../../../../defined-sets/ipv6-prefix-sets/prefix-set/name";

}

description

"reference to a prefix list to match the source address";

}

leaf destination-ipv6-prefix-list {

type leafref {

path "../../../../defined-sets/ipv6-prefix-sets/prefix-set/name";

}

description

"reference to a prefix list to match the destination address";

}

leaf protocol-set {

type leafref {

path "../../../../defined-sets/protocol-sets/protocol-set/name";

}

description

"reference to a protocol set to match the protocol field";

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches/ietf-acl:l4/ietf-acl:tcp" {

description

"Handle TCP flags and port sets.";

container flags-bitmask {

description

"Indicates how to handle TCP flags.";

uses tcp-flags;

}

leaf source-tcp-port-set {

type leafref {

path "../../../../defined-sets/port-sets/port-set/name";

}

description

"Reference to a port set to match the source port.";

}

leaf destination-tcp-port-set {

type leafref {

path "../../../../defined-sets/port-sets/port-set/name";

}

description

"Reference to a port set to match the destination port.";

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches/ietf-acl:l4/ietf-acl:udp" {

description

"Handle UDP port sets.";

leaf source-udp-port-set {

type leafref {

path "../../../../defined-sets/port-sets/port-set/name";

}

description

"Reference to a port set to match the source port.";

}

leaf destination-udp-port-set {

type leafref {

path "../../../../defined-sets/port-sets/port-set/name";

}

description

"Reference to a port set to match the destination port.";

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:matches/ietf-acl:l4/ietf-acl:icmp" {

description

"Handle ICMP type sets.";

leaf icmp-set {

type leafref {

path "../../../../defined-sets/icmp-type-sets/icmp-type-set/name";

}

description

"Reference to an ICMP type set to match the ICMP type field.";

}

}

augment "/ietf-acl:acls/ietf-acl:acl/ietf-acl:aces"

+ "/ietf-acl:ace/ietf-acl:actions" {

description

"rate-limit action.";

leaf rate-limit {

when "../ietf-acl:forwarding = 'ietf-acl:accept'" {

description

"rate-limit valid only when accept action is used.";

}

type decimal64 {

fraction-digits 2;

}

description

"Indicates a rate-limit for the matched traffic.";

}

}

}

<CODE ENDS>

[**6**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-6)**. Security Considerations (TBC)**

The YANG modules specified in this document define a schema for data

that is designed to be accessed via network management protocol such

as NETCONF [[RFC6241](https://datatracker.ietf.org/doc/html/rfc6241)] or RESTCONF [[RFC8040](https://datatracker.ietf.org/doc/html/rfc8040)]. The lowest NETCONF layer

is the secure transport layer, and the mandatory-to-implement secure

transport is Secure Shell (SSH) [[RFC6242](https://datatracker.ietf.org/doc/html/rfc6242)]. The lowest RESTCONF layer

is HTTPS, and the mandatory-to-implement secure transport is TLS

[[RFC8446](https://datatracker.ietf.org/doc/html/rfc8446)].

The Network Configuration Access Control Model (NACM) [[RFC8341](https://datatracker.ietf.org/doc/html/rfc8341)]

provides the means to restrict access for particular NETCONF or

RESTCONF users to a preconfigured subset of all available NETCONF or

RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are

writable/creatable/deletable (i.e., config true, which is the

default). These data nodes may be considered sensitive or vulnerable

in some network environments. Write operations (e.g., edit-config)

to these data nodes without proper protection can have a negative

effect on network operations. These are the subtrees and data nodes

and their sensitivity/vulnerability:

\* TBC

Some of the readable data nodes in this YANG module may be considered

sensitive or vulnerable in some network environments. It is thus

important to control read access (e.g., via get, get-config, or

notification) to these data nodes. These are the subtrees and data

nodes and their sensitivity/vulnerability:

\* TBC

[**7**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-7)**. IANA Considerations**

[**7.1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-7.1)**. URI Registration**

This document requests IANA to register the following URI in the "ns"

subregistry within the "IETF XML Registry" [[RFC3688](https://datatracker.ietf.org/doc/html/rfc3688)]:

URI: urn:ietf:params:xml:ns:yang:ietf-acl-enh

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

[**7.2**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-7.2)**. YANG Module Name Registration**

This document requests IANA to register the following YANG module in

the "YANG Module Names" subregistry [[RFC6020](https://datatracker.ietf.org/doc/html/rfc6020)] within the "YANG

Parameters" registry.

name: ietf-acl-enh

namespace: urn:ietf:params:xml:ns:yang:ietf-ietf-acl-enh

maintained by IANA: N

prefix: enh-acl

reference: RFC XXXX

[**8**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-8)**. References**

[**8.1**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-8.1)**. Normative References**

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", [BCP 14](https://datatracker.ietf.org/doc/html/bcp14), [RFC 2119](https://datatracker.ietf.org/doc/html/rfc2119),

DOI 10.17487/RFC2119, March 1997,

<<https://www.rfc-editor.org/rfc/rfc2119>>.

[RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](https://datatracker.ietf.org/doc/html/bcp81), [RFC 3688](https://datatracker.ietf.org/doc/html/rfc3688),

DOI 10.17487/RFC3688, January 2004,

<<https://www.rfc-editor.org/rfc/rfc3688>>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for

the Network Configuration Protocol (NETCONF)", [RFC 6020](https://datatracker.ietf.org/doc/html/rfc6020),

DOI 10.17487/RFC6020, October 2010,

<<https://www.rfc-editor.org/rfc/rfc6020>>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,

and A. Bierman, Ed., "Network Configuration Protocol

(NETCONF)", [RFC 6241](https://datatracker.ietf.org/doc/html/rfc6241), DOI 10.17487/RFC6241, June 2011,

<<https://www.rfc-editor.org/rfc/rfc6241>>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure

Shell (SSH)", [RFC 6242](https://datatracker.ietf.org/doc/html/rfc6242), DOI 10.17487/RFC6242, June 2011,

<<https://www.rfc-editor.org/rfc/rfc6242>>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language",

[RFC 7950](https://datatracker.ietf.org/doc/html/rfc7950), DOI 10.17487/RFC7950, August 2016,

<<https://www.rfc-editor.org/rfc/rfc7950>>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF

Protocol", [RFC 8040](https://datatracker.ietf.org/doc/html/rfc8040), DOI 10.17487/RFC8040, January 2017,

<<https://www.rfc-editor.org/rfc/rfc8040>>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration

Access Control Model", STD 91, [RFC 8341](https://datatracker.ietf.org/doc/html/rfc8341),

DOI 10.17487/RFC8341, March 2018,

<<https://www.rfc-editor.org/rfc/rfc8341>>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol

Version 1.3", [RFC 8446](https://datatracker.ietf.org/doc/html/rfc8446), DOI 10.17487/RFC8446, August 2018,

<<https://www.rfc-editor.org/rfc/rfc8446>>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair,

"YANG Data Model for Network Access Control Lists (ACLs)",

[RFC 8519](https://datatracker.ietf.org/doc/html/rfc8519), DOI 10.17487/RFC8519, March 2019,

<<https://www.rfc-editor.org/rfc/rfc8519>>.

[RFC8956] Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed.,

"Dissemination of Flow Specification Rules for IPv6",

[RFC 8956](https://datatracker.ietf.org/doc/html/rfc8956), DOI 10.17487/RFC8956, December 2020,

<<https://www.rfc-editor.org/rfc/rfc8956>>.

[**8.2**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#section-8.2)**. Informative References**

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams",

[BCP 215](https://datatracker.ietf.org/doc/html/bcp215), [RFC 8340](https://datatracker.ietf.org/doc/html/rfc8340), DOI 10.17487/RFC8340, March 2018,

<<https://www.rfc-editor.org/rfc/rfc8340>>.

[RFC8955] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M.

Bacher, "Dissemination of Flow Specification Rules",

[RFC 8955](https://datatracker.ietf.org/doc/html/rfc8955), DOI 10.17487/RFC8955, December 2020,

<<https://www.rfc-editor.org/rfc/rfc8955>>.

[RFC9132] Boucadair, M., Ed., Shallow, J., and T. Reddy.K,

"Distributed Denial-of-Service Open Threat Signaling

(DOTS) Signal Channel Specification", [RFC 9132](https://datatracker.ietf.org/doc/html/rfc9132),

DOI 10.17487/RFC9132, September 2021,

<<https://www.rfc-editor.org/rfc/rfc9132>>.

[**Appendix A**](https://datatracker.ietf.org/doc/html/draft-dbb-netmod-acl#appendix-A)**. Acknowledgements**

Many thanks to Jon Shallow and Miguel Cros for the discussion when

preparing this document.

Authors' Addresses

Oscar Gonzalez de Dios

Telefonica

Email: oscar.gonzalezdedios@telefonica.com

Samier Barguil

Telefonica

Email: samier.barguilgiraldo.ext@telefonica.com

Mohamed Boucadair

Orange

Email: mohamed.boucadair@orange.com