Extensions to the Access Control Lists (ACLs) YANG Model

draft-dbb-netmod-acl-03

O. Gonzalez de Dios, S. Barguil (**Telefonica**), M. Boucadair (**Orange**)

NETMOD WG Meeting

7th Nov 2022, IETF#115 @ London

Context Reminder

- RFC 8519 defines a YANG data model for Access Control Lists (ACLs)
  - Configuration of the **forwarding behaviour** in a device.
  - Definition of access-control-lists (ACLs), **entries** (ACEs), **matches,** and **actions**.

- We presented in IETF#112 a set of problems with the ACL YANG model as currently defined in RFC 8519

- We seeked in IETF#112 for the WG feedback about the following options:
  – *New version of the ACL model, minimizing non backwards compatible changes*

  Or

  – *Augmenting RFC 8519 in a new module. All existing structures are not touched*

Changes Since IETF#112

- Started to exercise the second option: that is, **augmentations** over RFC 8519

- draft-dbb-netmod-acl-03 proposes **a YANG module** to fix all the issues presented in IETF#112

3. Problem Statement & Gap Analysis . . . . . .
  3.1. Suboptimal Configuration: Lack of Support for 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 . . . . .

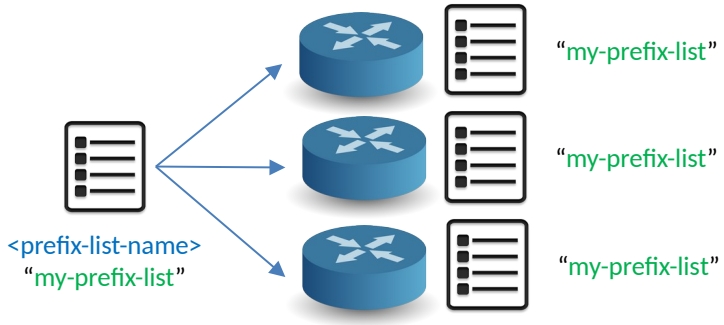
Samples are presented in the next slides

Manageability: Use of Defined sets (1)

- **Defined set:** reusable definition across several ACLs.
- Proposed defined sets:
  - **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).
  - **ICMP sets:** Uses to create lists of ICMP-based filters. This applies only when the protocol is set to ICMP or ICMPv6.
- Proposal:
  - **Augmentation** to add defined sets at acl level

```
augment /ietf-acl:acls/ietf-acl:acl:
+--rw defined-sets
```

  - **Augmentation** of matches to include a leaf-ref to the defined-set



Manageability: Use of Defined sets (2)

```
+--rw defined-sets
   |  +--rw ipv4-prefix-sets
   |  |  +--rw prefix-set* [name]
   |  |     +--rw name       string
   |  |     +--rw prefix*   inet:ip-prefix
   |  +--rw ipv6-prefix-sets
   |  |  +--rw prefix-set* [name]
   |  |     +--rw name       string
   |  |     +--rw 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
```

To create IPv4 *prefix lists*.

To create IPv6 *prefix lists*.

To create lists of TCP or UDP port values.

To create a list of protocols

To create lists of ICMP-based filters.

Additional Sets can be considered
(i.e Tags, MPLS Labels)

Handling of TCP Flags

- The augmented ACL structure includes a new leaf 'flags-bitmask' to better handle the TCP flags.
- Support matching operations as those supported in BGP Flow Spec
  - Simplifies operations and eases integration with other tools
  - The use of the bitmasks takes precedence of the old leaf in RFC8519

Proposal

```
{
      "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"
              }
            }]
          }
        }]
      }
    }
```

Handling of Fragments

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

Proposal

```
{
      "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"
                  }
                },
```

Rate-Limit Actions

- RFC8519 forwarding actions:
  - 'accept' (i.e., accept matching traffic),
  - 'drop' (i.e., drop matching traffic without sending any ICMP error message),
  - ‘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**.

- A new action called **"rate-limit"** is defined.

Proposal

```
{
        "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"
                }
              }]
            }
          }]
        }
      }
```

Seeking for WG Feedback

- Should we maintain the augmentation approach (as current -03 version) or switch to a bis approach?
  - The augmentation makes the structures less trivial to parse
  - The augmentation requires some conformance to be impose by normative language itself (e.g., which data node takes precedence)

- Where to position the defined sets?
  - Under “acls” in ACL module and leaf-ref in match in packet fields module
    - What happens if other modules import the packet match?
  - Standalone container in a new module
    - Easier to use by other modules should they require importing packet fields module
- Is this an item best worked in ***netmod wg***?

- Questions & Suggestions are welcome!!!!