YANG Data Model for Access Control List Configuration
draft-huang-netmod-acl-02

Lisa Huang, yihuan@cisco.com
Alexander Clemm, alex@cisco.com
Andy Bierman, andy@yumaworks.com

3/12/2013
Purpose

• **ACL: Access Control Lists**
  
  Used to filter traffic (“Firewall Rules”); major part of device configurations
  
  No configuration complete without ACLs

• **Why a YANG data model?**
  
  Netconf and YANG are intended for network device configuration
  
  Make ACL more accessible to automated applications, examples:
  
  - I2RS, Dynamic Intrusion Protection Systems
  
  Dynamic setup/configuration of services, e.g. temporary firewall rule adjustments for video conferences

• **Proposed model covers popular ACLs, incorporates rich set of filters**
  
  - IP ACL, MAC ACL, ARP ACL as initial ACL types
  
  - More than 50 filter criteria

• **ACL Configuration will benefit from standardization**
  
  Needed both by administrators and by applications
  
  Propose as standards-track Internet Draft
Actions taken from last IETF

• “How extensible” – could it support ACL chains, does it support multiple vendors
  ACL chains: New draft outlines which extensions would be needed; extensions are straightforward, can be incorporated into module itself if desired
  Engaged with Juniper colleagues; so far no objections/flags received

• “Are there models (non-IETF) that we can borrow, e.g. DMTF’s CIM”
  ACLs are not part of their standard but there are vendor extensions that cover
  Structure of those corresponds to similar design pattern; any mapping/mediation would appear straightforward

• “Engage IETF Security Experts”
  Message sent to IETF Security Directorate through Benoit (Ops Area Director)
  Engaged Joseph Salowey, comments:
    RFC 3588 – Diameter Protocol – defines AAA protocol IP filters against interfaces (not ACL)
      There are 3 parameters that are not part of current ACL data model
      1 could be added/extended if equivalent functionality desired (concerning direction)
      2 TCP filters, can be accommodated through TCP flag value and flag mask leaves
    Consider adding info who created an ACL rule – can be accommodated through special-purpose object
Draft-huang-netmod-acl-02.txt


- Changes from -01
  - Explained how to extend the current ACL to support ACL chain. Gave an example for ipv4. Same pattern can apply to ipv6, mac, and arp aces if needed.
  - Documented findings re: other models

- Modular and extensible ACL Management Framework

- 5 YANG modules
  - ACL – items common to ACLs regardless of type – "abstract superclass"
  - IP ACL, MAC ACL, ARP ACL as initial ACL types – can be extended
  - Common datatypes
    - Required by but not specific to ACL
    - Could be reused by other modules

- Emphasis on configuration
  - Very limited statistics
Proposal Summary

• Make YANG Data Model for ACL a standards-track working group item
  ACLs are an important part of device configurations
  Proprietary today
  Enabler for many applications, generally related to security
  Will clearly benefit from standardization

• Rev 02 of draft has already been posted
  Extensible + modular framework
  Includes support for 3 different types of ACL, more can be added
  Covers comprehensive set of parameters;
  feature statements allow for customization and device adaptation
  Possible items for discussion
    Inclusion of chaining feature
    Support for statistics
ACL concept

- **ACL**: Access Control List
  - An ordered set of rules used to filter traffic on a networking device

- **Access Control Entry (ACE)**: a representation of a rule
  - Left hand side: the matching criteria, or "filter"
  - Right hand side: the action to take – permit/deny a packet
  - Note: can generalize ACL with further actions: packet capture, audit logging, ...

- First rule that matches is applied
  - Most specific rules first to avoid rule shadowing

- **ACLs are applied against interfaces**
  - Interface refers to ACL (ACL specified independent of interface)
  - Different interfaces can use different ACLs, or use the same
ACL Types covered in the data model

- **IP ACLs**
  - Filter traffic based on IP information in the Layer 3 header of packets.

- **MAC ACLs**
  - Filter traffic using the information in the Layer 2 header of each packet.

- **ARP ACLs**
  - Filter IP- and non-IP-based traffic by checking the Ethernet type code field in the Layer 2 header.

Each ACL includes only ACEs of its type (no mix and match)

Framework can be extended with additional ACL types

Augment ACL YANG module
  - Follow design pattern of other ACL types, leverage common ACL data types
ACL module overview

Conceptual diagram only – for specific parameters and mapping to YANG data nodes refer to the draft.
**YANG Module Structure**

```yang
module: acl
  +--rw acls
    +--rw acl [name]
      |  +--rw name
      |  +--rw acl-type
      |      +--rw capture-session-id-global?
      |      +--rw (enable-match-counter-choices)?
      |      +--ro match?
    +--rw port-groups
      +--rw port-group [name]
        |  +--rw name
        |  +--rw groups
        +--rw timerange-group
          +--rw timerange-group [name]
            +--rw name
            +--rw time-ranges
            +--rw ip-address-groups
```

- **Generic ACL aspects, common to each ACL type**
- **Determines which types of ACEs can be inserted**
- **Not configuration related, could be separated**
- **Insertion point for specific ACL types (augmentation hook)**
- **Auxiliary convenience objects to simplify reuse of port groupings and schedule information** *(could move outside acls container)*
YANG module structure (contd.)

module: acl
  +--rw acls
     +--rw acl [name]
       | +--rw acl-ip:afi
       | +--rw acl-ip:ipv6-aces
       | +--rw acl-ip:ipv6-ace [name]
       |   +--rw acl-ip:name
       |   +--rw (remark-or-ipv6-case)?
       |     +--:(remark)
       |     | +--rw acl-ip:acl-remark
       |   +--:(ipv6-ace)
       |     | +--rw acl-ip:filters
       |     |   +-- filter parameters
       |     | +--rw acl-ip:actions
       |     |   +-- action parameters
       |   +-- ro acl-ip:match

Generics design pattern that is reflected in every ACL type
All ACL type specifics are in the filter parameters and in the actions

ACLs can include “comment lines” for human/admin consumption
Included in YANG module to maintain consistency with CLI

“left hand side”
“right hand side”
Not configuration related, could be separated

Indicates IP address type
**YANG module structure (contd.)**

module: acl
---rw acls
---rw acl [name]
  | ---rw acl-ip:afi
  | ---rw acl-ip:ipv4-aces
  |   | ---rw acl-ip:ipv4-ace [name]
  |   |     | ---rw acl-ip:name
  |   |     | ---rw (remark-or-ipv4-case)?
  |   |     |     | ---:(remark)
  |   |     |     |   | ---rw acl-ip:acl-remark
  |   |     |     |     | ---:(ipv4-ace)
  |   |     |     |   | ---rw acl-ip:filters
  |   |     |     |   |   | --- filter parameters
  |   |     |     |   |   | --- action parameters
  |   | --- ro acl-ip:match

---

**IPv4**

(IPv4 and IPv6 specified in same submodule)

- Indicates IP address type
- ACLs can include “comment lines” for human/admin consumption
- Included in YANG module to maintain consistency with CLI
- “left hand side”
- “right hand side”
- Not configuration related, could be separated

Generic design pattern that is reflected in every ACL type
All ACL type specifics are in the filter parameters and in the actions
YANG module structure (contd.)

module: acl
  +--rw acls
  +--rw acl [name]
    |  +--rw acl-mac:mac-aces
    |    |  +--rw acl-mac:mac-ace [name]
    |    |    +--rw acl-mac:name
    |    |    +--rw acl-mac:filters
    |    |    +-- acl-mac:actions
    |    |    +-- ro acl-mac:match
    |    +-- rw acl-mac:[name]
    |     +-- acl-mac:match
    |     +-- acl-mac:actions
    |     +-- acl-mac:filters
    |     +-- acl-mac:remark

MAC
(separate module)

Generic design pattern that is reflected in every ACL type
All ACL type specifics are in the filter parameters and in the actions
YANG module structure (contd.)

module: acl
  +--rw acls
  +--rw acl [name]
    |  +--rw acl-arp:arp-aces
    |   |  +--rw acl-arp:arp-ace [name]
    |   |     +--rw acl-arp:name
    |   |     +--rw (remark-or-arp-case)?
    |   |       +--:(remark)
    |   |       |  +--rw acl-arp:remark
    |   |       +--:(arp-ace)
    |   |       |  +--rw acl-arp:filters
    |   |       |     +-- filter parameters
    |   |       |     +--rw acl-arp:actions
    |   |       |         +-- action parameters
    |   |       +-- ro acl-arp:match

ARP
(separate module)

Generic design pattern that is reflected in every ACL type
All ACL type specifics are in the filter parameters and in the actions
module: acl
  +--rw acls
    +--rw acl [name]
      | +--rw acl-ip:ipv6-aces
      | | +--rw acl-ip:ipv6-ace [name]
      | | +--rw acl-ip:name
      | | +--rw (remark-or-ipv6-case)?
      | | | +--:(ipv6-ace)
      | | | +--rw acl-ip:filters
      | | | +-- rw (source-address-host-group)?
      | | | +-- rw (dest-address-host-goup)?
      | | | +-- rw acl-ip:protocol?
      | | | +-- rw acl-ip:capture-session-id?
      | | | +-- rw acl-ip:fragments?
      | | | +-- rw acl-ip:time-range?
      | | | +-- rw acl-ip:src-ports?
      | | | +-- rw acl-ip:dest-ports?
      | | | +--...
      | +--rw acl-ip: actions
      | | +-- rw acl-ip:action
      | | +-- rw acl-ip:log?

IPv6-specific parameters, but could add IP-v6 Specific filters

Insertion point for specific filters (augmentation hook)

Common actions but could add IP-specific actions later, such as copy, chain

Insertion point for additional actions, e.g. ACL chaining
ACL Chain ipv4 Example

augment "/acl:acls/acl:acl/acl-ip:ipv4-aces" +

"/acl-ip:ipv4-ace/acl-ip:actions" {

leaf chain {

typt acl-ref ;

description "Reference to another ACL name to chain the ACEs";

}

}
Q & A
Thank You
## Types specific to ACL (ACL module)

<table>
<thead>
<tr>
<th>YANG type</th>
<th>base type</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-comparator</td>
<td>enumeration</td>
</tr>
<tr>
<td>acl-action</td>
<td>enumeration</td>
</tr>
<tr>
<td>acl-remark</td>
<td>string</td>
</tr>
<tr>
<td>acl-type-ref</td>
<td>identityref</td>
</tr>
<tr>
<td>acl-ref</td>
<td>leafref</td>
</tr>
<tr>
<td>port-group-ref</td>
<td>leafref</td>
</tr>
<tr>
<td>ip-address-group-ref</td>
<td>leafref</td>
</tr>
<tr>
<td>time-range-ref</td>
<td>leafref</td>
</tr>
<tr>
<td>weekdays</td>
<td>bits</td>
</tr>
<tr>
<td>acl-name-string</td>
<td>string</td>
</tr>
</tbody>
</table>
### Common types – common module (required but not specific to ACL)

<table>
<thead>
<tr>
<th>YANG type</th>
<th>base type</th>
</tr>
</thead>
<tbody>
<tr>
<td>cos</td>
<td>uint8</td>
</tr>
<tr>
<td>tos</td>
<td>uint8</td>
</tr>
<tr>
<td>precedence</td>
<td>uint8</td>
</tr>
<tr>
<td>tcp-flag-type</td>
<td>enumeration</td>
</tr>
<tr>
<td>ether-type</td>
<td>string</td>
</tr>
<tr>
<td>ip-protocol</td>
<td>uint8</td>
</tr>
<tr>
<td>igmp-code</td>
<td>uint8</td>
</tr>
<tr>
<td>icmp-type</td>
<td>uint32</td>
</tr>
<tr>
<td>icmp-code</td>
<td>uint32</td>
</tr>
<tr>
<td>vlan-identifier</td>
<td>uint16</td>
</tr>
<tr>
<td>time-to-live</td>
<td>uint8</td>
</tr>
</tbody>
</table>
Example

• **ACL Example:**
  
  Denies TELNET traffic from 14.3.6.234 bound for host 6.5.4.1 from leaving.
  Denies all TFTP traffic bound for TFTP servers.
  Permits all other IP traffic.

• **ACL CLI:**

  access-list ip iacl
  
  deny tcp 14.3.6.234 0.0.0.0 host 6.5.4.1 eq 23
  deny udp any any eq tftp
  permit ip any any
XML instantiation

<acls>
  <acl>
    <name>iacl</name>
    <acl-type>ip-acl</acl-type>
    <enable-match-counter>false</enable-match-counter>
    <acl-ip:afi>ipv4</acl-ip:afi>
    <acl-ip:ipv4-aces>
      <acl-ip:ipv4-ace>
        <acl-ip:name>deny10</acl-ip:name>
        <acl-ip:filters>
          <acl-ip:ip-source-address>14.3.6.234</acl-ip:ip-source-address>
          <acl-ip:ip-source-mask>0.0.0.0</acl-ip:ip-source-mask>
          <acl-ip:ip-dest-host-address>6.5.4.1</acl-ip:ip-dest-host-address>
          <acl-ip:des-comparator>eq</acl-ip:des-comparator>
          <acl-ip:des-port>23</acl-ip:des-port>
        </acl-ip:filters>
        <acl-ip:actions>
          <acl-ip:action>deny</acl-ip:action>
        </acl-ip:actions>
      </acl-ip:ipv4-ace>
      ....
    </acl-ip:ipv4-aces>
  </acl>
</acls>
XML instantiation (contd.)

....

<acl-ip:ipv4-ace>
    <acl-ip:name>permit-20</acl-ip:name>
    <acl-ip:filters>
        <acl-ip:protocol>17</acl-ip:protocol>
        <acl-ip:ip-source-any/>
        <acl-ip:ip-dest-any/>
        <acl-ip:des-comparator>eq</acl-ip:des-comparator>
        <acl-ip:des-port>69</acl-ip:des-port>
    </acl-ip:filters>
    <acl-ip:actions>
        <acl-ip:action>deny</acl-ip:action>
    </acl-ip:actions>
</acl-ip:ipv4-ace>

<acl-ip:ipv4-ace>
    <acl-ip:name>any-30</acl-ip:name>
    <acl-ip:filters>
        <acl-ip:ip-source-any/>
        <acl-ip:ip-dest-any>
    </acl-ip:filters>
    <acl-ip:actions>
        <acl-ip:action>permit</acl-ip:action>
    </acl-ip:actions>
</acl-ip:ipv4-ace>
</acl-ip:ipv4-aces>
</acl>
</acls>