YANG based Config for MAC Privacy 802.1AEdk
Third Attempt

Don Fedyk (dfedyk@labn.net)
Outline

• Proto Config for MAC Privacy
• Moving configuration to standard language
Forward

• This presentation is for a discussion on detailed config.
• It may contain errors/omission and should be consider a work in progress.
• An updated version the presentation will be posted after discussion to correct it but it will remain a work in progress.
Instance Diagram for MACSec and MAC Privacy
MACsec and MAC Privacy

YANG Some lessons learned

• Instance Model – Where the YANG trees lives
• YANG Models – What to configure and what to display
  • Our bridge Model is a large superset that supports many permutations.
  • The model contains a lot of detail.
  • The tree provides a useful summary (a slice of the instance model)
• Validation
  • Pyang – validates a single model
  • Various other tools
• Instance Configuration – IEEE is in general only beginning to look at this
  • Yuma123
  • Confd (free version)
  • Yanglint (Used by IETF)
Validation versus Instance configuration

• Validation
  • YANG syntax is correct
  • YANG xpath is syntactically correct $x=y$ (but $x$ may be apples and $y$ may be oranges)
  • The whole set of permutations it the tree file or the xml description.

• Instance configuration
  • Config values are tested reference pointers are checked
  • YANG syntax is correct and multiple modules that are not related can exist side by side
  • $x = y$ and $x$ is the set of apple types and $y$ is a type of apple (Macintosh but not iphone!).
  • A slice of valid configuration references links are tested
MACsec and MAC Privacy

- Mainly 2 New Modules
  - `ieee802-dot1ae`
    - `ieee802-dot1ae-types`
  - `ieee802-dot1ae- pry`

Depends on:
- `ietf-yang-types`
- `ietf-inet-types`
- `iana-if-type`
- `ieee802-dot1q-bridge`
  - `ieee802-dot1q-types`
- `ieee802-dot1x`
  - `ieee802-dot1x-types`
- `ietf-interfaces`
- `ietf-system`
- `ieee802-types`
ieee802-dot1q-bridge.tree
Lots of stuff just by itself
Yanglint

- [github.com/CESNET/libyang](https://github.com/CESNET/libyang)
  - Parsing (and validating) schemas in YANG format.
  - Parsing (and validating) schemas in YIN format.
  - Parsing, validating and printing instance data in XML format.
  - Parsing, validating and printing instance data in JSON format (RFC 7951).
  - Manipulation with the instance data.
  - Support for default values in the instance data (RFC 6243).
  - Support for YANG extensions.
  - Support for YANG Metadata (RFC 7952).
  - yanglint - feature-rich YANG tool.
  - Current implementation covers YANG 1.0 (RFC 6020) as well as YANG 1.1 (RFC 7950).

- Loads multiple modules
- IETF uses for example config
  - Can test operational tree as well
  - Gives a Slice of the larger tree.
Configure a simple VLAN bridge.
Yanglint JSON output for a VLAN Bridge

```
data -t config -f json basic-vlan-bridge.xml
  "IEEE802-dot1q-bridge:bridges": {
    "bridge": [
      {
        "name": "bridge1",
        "address": "10-10-10-10-10-10",
        "bridge-type": "customer-vlan-bridge",
        "component": [
          {
            "name": "cv1",
            "id": 1,
            "type": "c-vlan-component"
          },
          {
            "name": "cv2",
            "id": 2,
            "type": "c-vlan-component"
          }
        ]
      }
    ]
  }

"IEEE802-dot1q-bridge:interfaces": {
  "interface": [
    {
      "name": "eth0",
      "type": "iana-if-type:bridge",
      "IEEE802-dot1q-bridge:bridge-port": {
        "bridge-name": "bridge1",
        "component-name": "cv1",
        "port-type": "c-vlan-bridge-port",
        "pvid": 1
      }
    },
    {
      "name": "eth1",
      "type": "iana-if-type:ethernetCsmacd",
      "IEEE802-dot1q-bridge:bridge-port": {
        "bridge-name": "bridge1",
        "component-name": "cv2",
        "port-type": "c-vlan-bridge-port",
        "pvid": 1
      }
    }
  ]
}
```

**Add MACsec**

Not too complicated
- No explicit VID tagging just PVID
Add MACsec (delta to interfaces)

Note partial file
Yanglint JSON output for a VLAN Bridge with MACsec

```
> data -t config -f json basic-vlan-bridge-with-macsec.xml
{
  "ieee802-dot1q-bridge:bridges": {
    "bridge": [ 
      {
        "name": "bridge1",
        "address": "10-10-10-10-10-10",
        "bridge-type": "customer-vlan-bridge",
        "component": [ 
          {
            "name": "cv1",
            "id": 1,
            "type": "c-vlan-component"
          },
          {
            "name": "cv2",
            "id": 2,
            "type": "c-vlan-component"
          }
        ]
      }
    ]
  },
  "ietf-interfaces:interfaces": {
    "interface": [ 
      {
        "name": "eth0",
        "type": "iana-if-type:bridge",
        "ieee802-dot1q-bridge:bridge-port": {
          "bridge-name": "bridge1",
          "component-name": "cv1",
          "port-type": "c-vlan-bridge-port",
          "pvid": 1
        },
        "ieee802-dot1x:pae": { 
          "pae-system": "pae1"
        }
      },
      {
        "name": "eth1",
        "type": "iana-if-type:ethernetCsmacd",
        "ieee802-dot1q-bridge:bridge-port": {
          "bridge-name": "bridge1",
          "component-name": "cv2",
          "port-type": "c-vlan-bridge-port",
          "pvid": 1
        },
        "ieee802-dot1x:pae": { 
          "pae-system": "pae1"
        }
      }
    ]
  },
  "ietf-system:system": {
    "contact": "test",
    "ieee802-dot1x:pae-system": { 
      "name": "pae1",
      "system-access-control": "enabled"
    }
  }
}
```

Note Abbreviated
TC table is 8 priorities 4 are shown etc
Yanglint JSON output for a VLAN Bridge with MACsec

- MACsec Position is controlled by the interface
- Interface position is controlled by the link to the component
- VLAN Behavior is controlled by the Bridge Port
Adding MAC Privacy is similar
Adding MAC Privacy is similar
Yanglint JSON output for a VLAN Bridge with MACsec

```json
> data -t config -f json basic-vlan-bridge-with-pry.xml
{
  "ieee802-dot1q-bridge:bridges": {
    "bridge": [  
      {  
        "name": "bridge1",
        "address": "10-10-10-10-10-10",
        "bridge-type": "customer-vlan-bridge",
        "component": [  
          {  
            "name": "cv1",
            "id": 1,
            "type": "c-vlan-component"
          },  
          {  
            "name": "cv2",
            "id": 2,
            "type": "c-vlan-component"
          }
        ]
      }  
    ],
    "ietf-interfaces:interfaces": {
      "interface": [  
        {  
          "name": "eth0",
          "type": "iana-if-type:bridge",
          "ieee802-dot1q-bridge:bridge-port": {  
            "bridge-name": "bridge1",
            "component-name": "cv1",
            "port-type": "c-vlan-bridge-port",
            "pvid": 1
          },
          "ieee802-dot1ae-pry:pry": {  
            "mac-privacy": "enabled",
            "user-priority-to-pry": [  
              {  
                "user-priority": 0,
                "privacy-type": "frame-a"
              },
              {  
                "user-priority": 1,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 2,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 3,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 4,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 5,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 6,
                "privacy-type": "express-channel"
              },
              {  
                "user-priority": 7,
                "privacy-type": "express-channel"
              }
            ],
            "privacy-channel": [  
              {  
                "pc": "standard-channel",
                "max-per-second-bitrate": 10000000000,
                "max-mppdu-size": 1500,
                "mppdu-priority": 3
              }
            ]
          }
        }
      ]
    }
  }
}
```

Note Abbreviated TC table is 8 priorities 4 are shown etc
Yanglint JSON output for a VLAN Bridge with MACsec

```json
{
    "user-priority": 3,
    "traffic-class": 3,
    "access-class-de0": 3,
    "access-class-de1": 3
}
```

```
"ieee802-dot1x:pae": {
    "pae-system": "pae1"
}
```

```
"ietf-system:system": {
    "contact": "test",
    "ieee802-dot1x:pae-system": {
        "name": "pae1",
        "system-access-control": "enabled"
    }
}
```
What about VLAN Control in single bridge

• Untagged ports can be tagged with PVID
• Tagged port can “flow through” if the TPID is of the same type.
• No Explicit VLAN tag control.
• Plenty of control plane control of filtering MACs and VIDs
What about EDEs?

• This is simple Bridge with a control plane controlling the VLANs
• The simple bridge does not cover the details of the EDE components
• Prior Presentations looked at Provider Bridged components
  • Conclusion was Map ports or handles to inner relay (PEP)
  • Then Incoming MAP inner relay to the Edge Component.
  • Edge compose applies outer VID
• Are there simple tagging configuration where a table could handle the bulk of EDE cases in the simple model?
  • Less flexibility than the complete Provider Bridge general model but good enough?
The rest of this presentation is for Discussion

• Is there a table format that can summarize the outer component for the simple EDE cases?
MACsec Config for EDEs

YANG models all these ( ) ports
Simple Cases we need

Detailed Model handled today

<table>
<thead>
<tr>
<th>data</th>
<th>Inner-TAG</th>
<th>SA</th>
<th>DA</th>
</tr>
</thead>
</table>

Incoming TAG

<table>
<thead>
<tr>
<th>data</th>
<th>Inner-TAG</th>
<th>SA</th>
<th>DA</th>
</tr>
</thead>
</table>

Possible Simple Model?

Table Mapping?

<table>
<thead>
<tr>
<th>data</th>
<th>Inner-TAG</th>
<th>SecTAG</th>
<th>SA</th>
<th>DA</th>
</tr>
</thead>
</table>

Outgoing TAGs

<table>
<thead>
<tr>
<th>data</th>
<th>Inner-TAG</th>
<th>SecTAG</th>
<th>Outer-TAG</th>
<th>SA</th>
<th>DA</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>data</th>
<th>SecTAG</th>
<th>Outer-TAG</th>
<th>SA</th>
<th>DA</th>
</tr>
</thead>
</table>
Cases we need: To be completed

<table>
<thead>
<tr>
<th>Incoming TAG</th>
<th>Inner TAG</th>
<th>Outer TAG</th>
<th>Needed ?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>C-TAG</td>
<td>S-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>C-TAG</td>
<td>C-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>S-TAG</td>
<td>S-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>C-TAG</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>S-TAG</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-TAG</td>
<td>Original C-TAG</td>
<td>S-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-TAG</td>
<td>Original C-TAG</td>
<td>C-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-TAG</td>
<td>Original C-TAG</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-TAG</td>
<td>Original S-TAG</td>
<td>S-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-TAG</td>
<td>Original S-TAG</td>
<td>C-TAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-TAG</td>
<td>Original S-TAG</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusions

• To be filled by meeting notes.