Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 11, 2018

Y. Dong
L. Xia
Huawei

September 07, 2017

The Data Model of Network Infrastructure Device Control Plane Security Baseline
draft-dong-sacm-nid-cp-security-baseline-00

Abstract

This document is one of the companion documents which describes the control plane security baseline YANG output for network infrastructure devices. The other parts of the whole document series [I-D.ietf- xia-sacm-nid-dp-security-baseline], [I-D.ietf-lin-sacm-nid-mp-security-baseline], [I-D.ietf-xia-sacm-nid-app-infr-layers-security-baseline] cover other parts of the security baseline for network infrastructure device in data plane, management plane, application layer and infrastructure layer respectively.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

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 March 11, 2018.

Table of Contents

1. Introduction

1.1. Objective

Nowdays network infrastructure devices such as switches, routers, and firewalls are always under the attack of the well-known network security threats which are sammrized in [I-D.ietf-xia-sacm-dp-security-profile]. Hence it is significant to ensure that the devices in a specific network meet the minimal security requirements according to their intended functions. In this case, the concept of security baseline for the network infrastructure device has been proposed in the above mentioned draft [I-D.ietf-xia-sacm-dp-security-profile] as well. The security baseline refers to the basic and compulsory capabilities of identifying the possible threats and vulnerabilities in the device itself, and enfocing the security hardening measurement. And it could be set to benchmark the security posture of an individual network device.

Basically, the overall security baseline of a particular network infrastructure device can be designed and deployed into three different layers, namely the application layer, the network layer, and the infrastructure layer. Moreover, the network layer security baseline is further classified into data plane, control plane, and management plane. In this document, we focus on the designation of data model for control plane security baseline while the security baseline of other layers and planes are proposed in the companion documents.

The control plane security basedline focus on the control signaling security of the network infrastructure device. The aim is to protect the normal information exchange between devices against various attcks (i.e. eavesdropping, tampering, spoofing and flooding attack) and restrict the malicious control signaling, for ensuring the correct network topology and forwading behavior.

1.2. Security Baseline Data Model Design

The security baseline of a certain device is dependent on many factors including but not limited to the different device types (i.e., router, switch, firewall) and their corresponding security features supported, and the specific security requirements of network operators. Owning to such a number of variations, it is impossible to design a comprehensive set of baseline for all devices. This document and the companion ones are going to propose the most important and universal points of them. More points can be added in future following the data model scheme specified in this document.

[I-D.ietf-birkholz-sacm-yang-content] defines a method of constructing the YANG data model scheme for the security posture assessment of the network infrastructure device by brokering of YANG push telemetry via SACM statements. The basic steps are:

- use YANG push mechanism[I-D.ietf-netconf-yang-push]to collect the created streams of notifications (telemetry) [I-D.ietf-netconf-subscribed-notifications]providing SACM content on SACM data plane, and the filter expressions used in the context of YANG subscriptions constitute SACM content that is imperative guidance consumed by SACM components on SACM management plane;

- then encapsulate the above YANG push output into a SACM Content Element envelope, which is again encapsulated in a SACM statement envelope;

- lastly, publish the SACM statement into a SACM domain via xmpp-grid publisher.

In this document, we follow the same way as [I-D.ietf-birkholz-sacm-yang-content] to define the YANG output for network infrastructure device security baseline posture based on the SACM information model definition [I-D.ietf-sacm-information-model].

1.3. Summary

The following contents propose part of the security baseline YANG output for network infrastructure device: control plane security baseline. The companion documents [I-D.ietf- xia-sacm-nid-dp-security-baseline], [I-D.ietf-lin-sacm-nid-mp-security-baseline], [I-D.ietf-xia-sacm-nid-app-infr-layers-security-baseline] cover other parts of the security baseline YANG output for network infrastructure device respectively: control plane security baseline, management plane security baseline, application layer and infrastructure layer security baseline.

2. Terminology

2.1. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2.2. Definition of Terms

This document uses the terms defined in [I-D.draft-ietf-sacm-terminology].

3. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node and "*" denotes a "list" and "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- o Ellipsis ("...") stands for contents of subtrees that are not shown.

4. Data Model Structure

A large amount of control protocols such as the typical TCP/IP stack and BGP in the control plane of network infrastructure device provide many operational services (i.e. farwording behavior control). These control protocols could be either the target under attack or the medium to attack the devices. The security baseline of several widely used protocols are specified in this section.

4.1. BGP

In a BGP network, TCP is always selected as the transport layer protocol. Thus it always subject to most of the attacks that targeting TCP-based protocols. In order to secure the BGP network, three types of functions, namely the GTSM, the RPKI, and the BGP peer connection authentication, could be configured in network device. This section specifies the authentication and RPKI configurations. The GTSM is summarized in another individual section together with some other protocols that all supports GTSM.

Various kinds of authentication techniques are able to be used for securing the TCP connections between BGP neighbors. They only allows the authorized peers to establish neighbor relationship with local device so that the information exchanged between the BGP neighbors via the TCP connection cannot be altered.

The Resource Public Key Infrastructure (RPKI) is usually applied in a network equipt with a RPKI server to secure the inter-domain BGP routing. The device is required to establish a connection to the RPKI server and then downloads or updates the Route Origin Authorizations (ROAs), which links certain IP prefixes or prefix range with an autonomous system (AS), from the RPKI server. After that, the received BGP route information is validated against the downloaded/updated ROAs to verify whether the BGP prefixe originates from the expected AS.

```
 module: bgp-sec-config
     +--rw bgp-rpki
     |  +--rw bgp-rpki-session-config* [session-ipv4-addr]
     |  |  +--rw session-ipv4-addr        ipv4-address
     |  |  +--rw port-number              unit16
     |  |  +--rw cipher-password?         string
     |  |  +--rw aging-time?              unit32
     |  |  +--rw refresh-time?            unit16
     |  |  +--rw rpki-limit?
```

```
|  |     +--rw limit                 unit32
|  |     +--rw (action-type)
|  |        +--:(alert)
|  |        |  +--rw enable          boolean
|  |        +--:(idle-forever)
|  |        |  +--rw enable          boolean
|  |        +--:(idle-timeout)
|  |           +--rw timeout         unit16
|  +--rw origin-as-validate-utilization
|     +--rw origin-validation-enable boolean
|     +--rw origin-as-validate       boolean
|     +--rw allow-invalide           boolean
|     +--rw (peer-identification-method)
|        +--:(group)
|        |  +--rw peer-group* [group-name]
|        |     +--rw group-name      string
|        |     +--rw advertise-enable boolean
|        +--:(ip)
|           +--rw peer-ip* [ipv4-addr]
|              +--rw ipv4-addr     ipv4-address
|              +--rw advertise-enable boolean
|
+--rw bgp-authentication* [bgp-as-number]
   +--rw bgp-as-number             unit16
   +--rw (peer-identification-method)
      +--:(group)
      |  +--rw peer-group* [group-name]
      |     +--rw group-name       string
      |     +--rw (authentication-method)
      |        +--:(md5)
      |        |  +--rw password-type:{plain|cipher} enumeration
      |        |  +--rw password-text  string
      |        +--:(keychain)
      |           +--rw keychain-name
      +--:(ip)
         +--rw peer-ip* [ipv4-addr]
            +--rw ipv4-addr        inet-type:ipv4-address
            +--rw (authentication-method)
               +--:(md5)
               |  +--rw password-type:{plain|cipher} enumeration
               |  +--rw password-text  string
               +--:(keychain)
                  +--rw keychain-name  string
```

4.2. OSPF

There are a number of ways for spoofing procotol packet to attack OSPF protocol. One possible scenario is that the rogue device inject manipulated routing information to cause a Denial-of-Service attack.

Authentication has been demonstrated as a powerful tool to identify and drop these spoofing packets to protect OSPF protocol and secure the connection between the OSPF neighbors. A widely range of authentication methods can be deployed in a network device such as MD5, HAMC-MD5, and keychain. As shown in the following tree diagram, the authentication can be deployed in either area or interface basis.

```
 module:ospf-sec-config
     +--rw ospf-authentication
        +--rw area-authentication* [area-id]
        |  +--rw area-id              unit16
        |  +--rw (authentication-method)
        |     +--:(simple-authen)
        |     |  +--rw password-type:{plain|cipher} enumeration
        |     |  +--rw password-text  string
        |     +--:(md5-hmac-authen)
        |     |  +--rw sub-mode:
        |     |  |      {md5|hmac-md5|hmac-sha256} enumeration
        |     |  +--rw password-type   enumeration
        |     |  +--rw password-text   string
        |     +--:(keychain-authen)
        |        +--rw keychain-name   string
        +--rw interface-authentication* [interface-number]
           +--rw interface-type        enumeration
           +--rw interface-number      unit8
           +--rw (authentication-method)
              +--:(simple-authen)
              |  +--rw password-type   enumeration
              |  +--rw password-text   string
              +--:(md5-hmac-authen)
              |  +--rw sub-mode        enumeration
              |  +--rw password-type   enumeration
              |  +--rw password-text   string
              +--:(keychain-authen)
                 +--rw keychain-name   string
```

4.3. IS-IS

IS-IS optional checksum function adds the a checksum TLV in SNP and hello packet. The device firstly check the correctness of checksum TVL when it receive the packet. It secure the data in data link layer.

IS-IS authentication encapsulate the authentication information in hello packet, LSP packet, and SNP packet. Only the packets passed the verification will be further processed. The IS-IS authentication is mainly used to secure packet in network layer.

```
    module:isis-sec-config
        +--rw isis-optional-checksum
        |  +--rw enable                    boolean
        +--rw isis-authentication
           +--rw area-authentication* [process-id]
           |  +--rw process-id          unit32
           |  +--rw (authentication-method)
           |     +--:(simple)
           |     |  +--rw authen-password-mode:{op|osi} enumeration
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--:(md5)
           |     |  +--rw authen-password-mode:{op|osi} enumeration
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--:(keychain)
           |     |  +--rw keychain-name                 string
           |     +--:(hmac-sha256)
           |     |  +--rw key-id                        unit16
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--rw snp-packet:
           |     |      {authentication-avoid|send-only} enumeration
           |     +--rw all-send-only?                   boolean
           +--rw domain-authentication* [process-id]
           |  +--rw process-id                      unit32
           |  +--rw (authentication-method)
           |     +--:(simple)
           |     |  +--rw authen-password-mode:{op|osi} enumeration
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--:(md5)
           |     |  +--rw authen-password-mode:{op|osi} enumeration
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--:(keychain)
           |     |  +--rw keychain-name                 string
           |     +--:(hmac-sha256)
           |     |  +--rw key-id                        unit16
           |     |  +--rw password-type:{plain|cipher}  enumeration
           |     |  +--rw password-text                 string
           |     +--rw snp-packet                       enumeration
           |     +--rw all-send-only?                   boolean
           +--rw interface-authentication* [interface-number]
              +--rw interface-type                  enumeration
              +--rw interface-number                pub-type:ifNum
              +--rw (authentication-method)
                 +--:(simple)
```

```
                |  +--rw authen-password-mode:{op|osi} enumeration
                |  +--rw password-type:{plain|cipher}  enumeration
                |  +--rw password-text                 string
                +--:(md5)
                |  +--rw authen-password-mode:{op|osi} enumeration
                |  +--rw password-type:{plain|cipher}  enumeration
                |  +--rw password-text                 string
                +--:(keychain)
                |  +--rw keychain-name                 string
                +--:(hmac-sha256)
                |  +--rw key-id                        unit16
                |  +--rw password-type:{plain|cipher}  enumeration
                |  +--rw password-text                 string
                +--rw authen-level?:{level1|level2} enumeration
                +--rw send-only?                    boolean
```

4.4. MPLS

RSVP authentication is suggested to configure in the device in order to improve the network security and protect the local device against the malicious attack. It prevent the establishment of illegal RSVP peer connection in the following situation

The peer was unauthorized to establish connection with local device;

The attacker establish connection with locol device via spoofing RSVP packet.

Furthermore, it introduce a few enhancement to verify the lifetime, handshake and message window size for protection of RSVP against the playback attack and the termination of authentication relationships caused by packet out of order problem.

As shown in the tree diagram, the LDP also support MD5 and keychain authentication.

```
  module:mpls-sec-config
     +--rw rsvp-sec-config
     |  +--rw rsvp-authentication
     |     +--rw interface-authentication
     |     |  +--rw interface-authen* [interface-number]
     |     |     +--rw interface-type                 enumeration
     |     |     +--rw interface-number               pub-type:ifNum
     |     |     +--rw (authentication-method)
     |     |        +--:(md5)
     |     |        |  +--rw password-type:{plain|cipher} enumeration
     |     |        |  +--rw password-text                string
```

```
|     |        +--:(keychain)
|     |        |  +--rw keychain-name                string
|     |        +--rw life-time?              yang-type:timestamp
|     |        +--rw handshake-enable?               boolean
|     |        +--rw window-size?                    unit8
|     +--rw peer-authentication
|        +--rw peer-authen* [peer-addr]
|           +--rw peer-addr                    inet-type:ip-address
|           +--rw (authentication-method)
|              +--:(md5)
|              |  +--rw password-type:{plain|cipher} enumeration
|              |  +--rw password-text                string
|              +--:(keychain)
|              |  +--rw keychain-name                string
|              +--rw challenge-maximum-miss-times?   unit8
|              +--rw challenge-retrans-interval?     unit16
|              +--rw life-time?              yang-type:timestamp
|              +--rw handshake-enable?               boolean
|              +--rw window-size?                    unit8
+--rw ldp-sec-config
   +--rw ldp-authentication
   +--rw (authentication-method)
      +--:(keychain)
      |  +--rw (authen-object)
      |     +--:(peer-single)
      |     |  +--rw single-peer-authen* [peer-id]
      |     |     +--rw peer-id                 dotted decimal
      |     |     +--rw keychain-name               string
      |     +--:(peer-group)
      |     |  +--rw group-peer-authen* [ip-prefix-name]
      |     |     +--rw ip-prefix-name              string
      |     |     +--rw keychain-name               string
      |     +--:(peer-all)
      |        +--rw keychain-name                  string
      |        +--rw exclude-peer-id?            dotted decimal
      +--:(md5)
         +--rw (authen-object)
            +--:(peer-single)
            |  +--rw single-peer-authen* [peer-lsr-id]
            |     +--rw peer-lsr-id                dotted decimal
            |     +--rw password-type:{plain|cipher} enumeration
            |     +--rw password-text              string
            +--:(peer-group)
            |  +--rw group-peer-authen* [ip-prefix-name]
            |     +--rw ip-prefix-name             string
            |     +--rw password-type:{plain|cipher} enumeration
            |     +--rw password-text              string
            +--:(peer-all)
```

```
                  +--rw password-type:{plain|cipher}  enumeration
                  +--rw password-text                 string
```

4.5. Keychain

Authentication is a widely used technique to ensure the packet information are not been changed/altered by attackers. It requires the information sender and receiver to share the authentication information including the key and algorithm. In addition, the key pairs cannot be delivered in the network (symmetric). However, in order to improve the its reliability, the encryption algorithm and the keys have to be renewed dynamically. It is a complicated and time consuming process to change the keys and algorithm for all the used protocols manually. The keychain provide an solution to renew the authentication keys and algorithm periodically in a dynamic fashion.

```
module:keychain-config
    +--rw keychain-config* [keychain-name]
       +--rw keychain-name               string
       +--rw keychain-mode:
       |     {absolute|periodic|daily|
       |      weekly|monthly|yearly}    enumeration
       +--rw receive-tolerance?
       |  +--:(finite)
       |  |  +--rw tolerance-value              unit16
       |  +--:(infinite)
       |     +--rw infinite-enable              boolean
       +--rw time-mode:{utc|lmt}                enumeration
       +--rw digest-length?                     boolean
       +--rw keychain-id* [key-id]
          +--rw key-id                          unit8
          +--rw keychain-string-type:{plain|cipher} enumeration
          +--rw keychain-string-text            string
          +--rw keychain-algorithm:
          |    {hmac-md5|hmac-sha-256|hmac-sha1_12|
          |     hmac-sha1_20|md5|sha-1|sha-256} enumeration
          +--rw default-key-id?                  unit8
          +--rw (send-time-mode)
          |  +--:(absolute)
          |  |  +--rw start-time           yang-type:timestamp
          |  |  +--rw start-date         yang-type:date-and-time
          |  |     +--rw (count-type)
          |  |        +--:(duration)
          |  |        |  +--rw (finite-or-infinite)
          |  |        |     +--:(finite)
          |  |        |     |  +--rw duration-value  unit32
          |  |        |     +--:(infinite)
```

```
|  |        |        +--rw infinite-enable boolean
|  |        +--:(end)
|  |           +--rw end-time    yang-type:timestamp
|  |           +--rw end-date    yang-type:date-and-time
|  +--:(periodic-daily)
|  |  +--rw start-time           yang-type:timestamp
|  |  +--rw end-time             yang-type:timestamp
|  +--:(periodic-weekly)
|  |  +--rw (count-type)
|  |     +--:(continues)
|  |     |  +--rw start-day-name  enumeration
|  |     |  +--rw end-day-name    enumeration
|  |     +--:(discrete)
|  |        +--rw day-name*       enumeration
|  +--:(periodic-montly)
|  |  +--rw (count-type)
|  |     +--:(continues)
|  |     |  +--rw start-date    yang-type:date-and-time
|  |     |  +--rw end-date      yang-type:date-and-time
|  |     +--:(discrete)
|  |        +--rw date*         yang-type:date-and-time
|  +--:(periodic-yearly)
|     +--rw (count-type)
|        +--:(continues)
|        |  +--rw start-month     enumeration
|        |  +--rw end-month       enumeration
|        +--:(discrete)
|           +--rw month*          enumeration
+--rw (receive-time-mode)
   +--:(absolute)
   |  +--rw start-time            yang-type:timestamp
   |  +--rw start-date          yang-type:date-and-time
   |     +--rw (count-type)
   |        +--:(duration)
   |        |  +--rw (finite-or-infinite)
   |        |     +--:(finite)
   |        |     |  +--rw duration-value  unit32
   |        |     +--:(infinite)
   |        |        +--rw infinite-enable boolean
   |        +--:(end)
   |           +--rw end-time    yang-type:timestamp
   |           +--rw end-date  yang-type:date-and-time
   +--:(periodic-daily)
   |  +--rw start-time          yang-type:timestamp
   |  +--rw end-time            yang-type:timestamp
   +--:(periodic-weekly)
   |  +--rw (count-type)
   |     +--:(continues)
```

```
|     |  +--rw start-day-name   enumeration
|     |  +--rw end-day-name     enumeration
|     +--:(discrete)
|        +--rw day-name*        enumeration
+--:(periodic-montly)
|  +--rw (count-type)
|     +--:(continues)
|     |  +--rw start-date    yang-type:date-and-time
|     |  +--rw end-date      yang-type:date-and-time
|     +--:(discrete)
|        +--rw date*         yang-type:date-and-type
+--:(periodic-yearly)
   +--rw (count-type)
      +--:(continues)
      |  +--rw start-month    enumeration
      |  +--rw end-month      enumeration
      +--:(discrete)
         +--rw month*         enumeration
```

4.6. GTSM

Attackers send a large amount of forging packets to a target network device. Then the forging packets are delivered to the cpu straigtforward when the destinations are correctly checked. The CPU will be overloaded owning to processing such a number of protocol packets. In order to protect the CPU against the CPU utilization attack, a GTSM (generized TTL security mechanism) function is configured to check the TTL (time to live) in the IP head. The packets will send to cpu for further processing only if the TTL number is whithin a pre-defined range.

As shown in the three diagram in the following figure, the GTSM function is configured separately for individual procotols. Each of the protocols, even each list instances in a protocol, has its own pre-defined TTL range.

```
module:gtsm
    +--rw gtsm-config
       +--rw default-gtsm-action:{drop|pass}
       +--rw bgp-gtsm* [bgp-as-number]
       |  +--rw bgp-as-number             unit32
       |  +--rw (peer-identification-method)
       |     +--:(group)
       |     |  +--rw peer-group* [group-name]
       |     |     +--rw group-name       string
       |     |     +--rw valid-ttl-hops   unit16
       |     +--:(ip)
       |        +--rw peer-ip*  [ipv4-addr]
       |           +--rw ipv4-addr   inet-type:ipv4-address
       |           +--rw valid-ttl-hops   unit8
       +--rw ospf-gtsm* [vpn-instance-name]
       |  +--rw vpn-instance-name       string
       |  +--rw valid-ttl-hops           unit16
       +--rw mpls-ldp-gtsm* [peer-ip-addr]
       |  +--rw peer-ip-addr          inet-type:ip-address
       |  +--rw valid-ttl-hops           unit16
       +--rw rip-gtsm* [vpn-instance-name]
          +--rw vpn-instance-name?      string
          +--rw valid-ttl-hops          unit16
```

5. Network Infrastructure Device Security Baseline Yang Module

TBD

6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

TBD.

8. Acknowledgements

TBD

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[I-D.ietf-netconf-subscribed-notifications] Voit, E., Clemm, A., Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Custom Subscription to Event Notifications", draft-ietf-netconf-subscribed-notifications-03 (work in progress), July 2017.

[I-D.ietf-netconf-yang-push] Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", draft-ietf-netconf-yang-push-08 (work in progress), August 2017.

[I-D.ietf-sacm-information-model] Waltermire, D., Watson, K., Kahn, C., Lorenzin, L., Cokus, M., Haynes, D., and H. Birkholz, "SACM Information Model", draft-ietf-sacm-information-model-10 (work in progress), April 2017.

Authors' Addresses

Yue Dong
Huawei

Email: dongyue6@huawei.com

Liang Xia
Huawei

Email: frank.xialiang@huawei.com

SACM Working Group
Internet-Draft
Intended status: Informational
Expires: January 22, 2020

S. Banghart
D. Waltermire
NIST

July 21, 2019

Definition of the ROLIE Software Descriptor Extension draft-ietf-sacm-rolie-softwaredescriptor-08

Abstract

This document uses the "information-type" extension point as defined in the Resource-Oriented Lightweight Information Exchange (ROLIE) [RFC8322] Section 7.1.2 to better support Software Record and Software Inventory use cases. This specification registers a new ROLIE information-type, "software-descriptor", that allows for the categorization of information relevant to software description activities and formats. In particular, the usage of the ISO 19770-2:2015 Software Identification Tag (SWID Tag) and the Concise SWID (COSWID) formats in ROLIE are standardized. Additionally, this document discusses requirements and usage of other ROLIE elements in order to best syndicate software description information.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

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 January 22, 2020.

Copyright Notice

Table of Contents

1. Introduction

This document defines an extension to the Resource-Oriented Lightweight Information Exchange (ROLIE) [RFC8322] to support the publication of software descriptor information. Software descriptor information is information that characterizes static software components, packages, and installers; including identification, version, software creation and publication, and file artifact information.

Software descriptor information provides data about what might be installed, but doesn't describe a specific software installation's

configuration or execution. This static approach to software description is a tightly limited scope that still covers the majority of current use cases for software inventory and record keeping.

Some possible use cases for software descriptor information ROLIE Feeds (Section 6.1 of [RFC8322]) include:

- Software providers can publish software descriptor information so that software researchers, enterprises, and users of software can understand the collection of software produced by that software provider.

- Organizations can aggregate and syndicate collections of software descriptor information provided by multiple software providers to support software-related analysis processes (e.g., vulnerability analysis) and to provide downsteam services (e.g., software configuration checklist repositories).

- End user organizations can consume software descriptor information along with related software vulnerability and configuration information to provide the data needed to automate software asset, patch, and configuration management practices.

- Organizations can use software descriptors to support verification of other entities through integrity measurement mechanisms.

This document supports these use cases by describing the content requirements for Feeds and Entries of software descriptor information that are to be published to or retrieved from a ROLIE repository.

2. Terminology

The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

As an extension of [RFC8322], this document refers to many terms defined in that document. In particular, the use of "Entry" and "Feed" are aligned with the definitions presented in RFC8322.

Several places in this document refer to the "information-type" of a Resource (Entry or Feed). This refers to the "term" attribute of an "atom:category" element whose scheme is "urn:ietf:params:rolie:category:information-type". For an Entry, this value can be inherited from it’s containing Feed as per [RFC8322].

3. Background

In order to effectively protect and secure an endpoint, it is vital to know what the software load of that endpoint is. Software load, the combination of software, patches and installers on a device, represents a significant portion of the endpoint's attack surface. Unfortunately, without a reliable and secure package manager, or a secured and managed operating system with strict software whitelisting, tracking what software is installed on an endpoint is currently not feasible without undue effort. Even attempting to whitelist software is difficult without a way of identifying software and its editions, versions and hotfixes.

Software descriptor information, such as that standardized in the ISO 19770-2:2015 Software Identification Tag (SWID) format or expressed in proprietary enterprise databases, attempts to provide as much data about this software as possible.

Once this information is expressed, it needs to be stored and shared to internal and external parties. ROLIE provides a mechanism to handle this sharing in an automation-friendly way.

4. The "software-descriptor" information type

When an "atom:category" element has a "scheme" attribute equal to "urn:ietf:params:rolie:category:information-type", the "term" attribute defines the information type of the associated resource. A new valid value for this "term": "software-descriptor", is described in this section and registered in Section 8.1. When this value is used, the resource in question is considered to have an information-type of "software-descriptor" as per [RFC8322] Section 7.1.2.

The "software-descriptor" information type represents any static information that describes a piece of software. This document uses the definition of software provided by [RFC4949]. Note that as per this definition, this information type pertains to static software, that is, code on the disc. The "software-descriptor" information type is intended to provide a category for information that does one or more of the following:

identifies and characterizes software: information that provides quantative and qualitative data describing software. This information identifies and charaterizes a given instance of software.

provides software installer metadata: information about software used to install other software. This metadata identifies, and characterizes a software installation package or media.

describes stateless installation metadata: information that describes the software post-deployment, such as files that may be deployed during an installation. It is expected that this metadata is produced generally for a given installation, and may not exactly match the actual installed files on a given endpoint.

Provided below is a non-exhaustive list of information that may be considered to be of a software-descriptor information type.

- Naming information: IDs and names that aid in the identification of a piece of software
- Version and patching information: Version numbers, patch identifiers, or other information that relates to software updates and patches.
- Vendor and source information: Includes where the software was developed or distributed, as well as where the software installation media may be located.
- Payload and file information: information that describes or enumerates the files and folders that make up the piece of software, and information about those files.
- Descriptive information and data: Any information that otherwise characterizes a piece of software, such as libraries, runtime environments, target operating systems, intended purpose or audience, etc.

It is important to note that software descriptor information is static for a given piece of software. That is, the information expressed is the data that doesn't change from the publication of the software to its final install. Information about the current status (e.g. install location, memory usage, CPU usage, launch parameters, job progress, etc.), is out of scope of this information type.

5. rolie:property Extensions

This document registers new valid rolie:property names as follows:

5.1. urn:ietf:params:rolie:property:swd:swname

This property provides an exposure point for the plain text name of the software being described. Naming of software is not a well standardized process, and software names can change between product versions or editions. As such, care should be taken that this value is set as consistently as possible by generating it directly from an attached software descriptor resource.

5.2. urn:ietf:params:rolie:property:swd:swversion

This property provides an exposure point for the version of the software being described. This value should be generated or taken from the software descriptor linked to by the entry. This helps avoid, but does not prevent, inconsistent versioning schemes being shared.

5.3. urn:ietf:params:rolie:property:swd:swcreator

This property provides an exposure point for a plain text name of the creator of the software being described. This is in many cases an organization or company, but certainly could be a single person. Most software descriptor formats include this information, and where possible, this property should be set equal to that value.

6. Data format requirements

This section defines usage guidance and additional requirements related to data formats above and beyond those specified in [RFC8322]. The following formats are expected to be commonly used to express software descriptor information. For this reason, this document specifies additional requirements to ensure interoperability.

6.1. The ISO SWID 2015 format

6.1.1. Description

ISO/IEC 19770-2:2015 defines a software record data format referred to as a "SWID Tag". It provides several tag types:

- primary: provides descriptive and naming information about software,
- patch: describes non-standalone software meant to patch existing software,
- corpus: describes the software installation media that installs a given piece of software,
- supplemental: provides additional metadata to be deployed alongside a tag.

For a more complete overview as well as normative requirements, refer to ISO/IEC 19770-2:2015 [SWID].

For additional requirements and guidance around creation of SWID Tags, consult NIST Internal Report 8060 [NISTIR8060].

6.1.2. Requirements

For an Entry to be considered as a "SWID Tag Entry", it MUST fulfill the following conditions:

- The information-type of the Entry is "software-descriptor". For a typical Entry, this is derived from the information type of the Feed it is contained in. For a standalone Entry, this is provided by an "atom:category" element.

- The document linked to by the "href" attribute of the "atom:content" element is a 2015 SWID Tag per ISO/IEC 19770-2:2015.

A "SWID Tag Entry" MUST conform to the following requirements:

- The value of the "type" attribute of the "atom:content" element MUST be "application/xml".

- There MUST be one "rolie:property" with the "name" attribute equal to "urn:ietf:params:rolie:property:content-id" and the "value" attribute exactly equal to the "<tagid>" element in the attached SWID Tag. This allows ROLIE consumers to more easily search for SWID tags without needing to download the tag itself.

- There MUST be one "rolie:property" with the "name" attribute equal to "urn:ietf:params:rolie:property:swd:swname", and the "value" attribute equal to the value of the "<name>" element in the attached SWID Tag. As above, this helps ROLIE consumers search and filter Entries.

- There MAY be a property element with the "name" attribute equal to "urn:ietf:params:rolie:property:swd:swversion". When this property appears, its value MUST be equal to the value of the "version" element in the attached SWID Tag.

6.2. The Concise SWID format

6.2.1. Description

The Concise SWID (COSWID) format is an alternative representation of the SWID Tag format using a Concise Binary Object Representation (CBOR) encoding. CBOR provides the format with a reduced size that is more suitable for constrained devices. COSWID provides the same features and attributes as are specified in ISO 19770-2:2015, plus:

- a straight forward method to sign and encrypt using COSE, and
- additional attributes that provide an improved structure to include file hashes intended to be used as Reference Integrity Measurements (RIM).

For more information and the complete specification, refer to the COSWID internet draft [I-D.ietf-sacm-coswid].

6.2.2. Requirements

For an Entry to be considered as a "COSWID Tag Entry", it MUST fulfill the following conditions:

- The information-type of the Entry is "software-descriptor". For a typical Entry, this is derived from the information-type of the Feed it is contained in. For a standalone Entry, this is provided by an "atom:category" element.
- The document linked to by the "href" attribute of the "atom:content" element is a COSWID Tag per [I-D.ietf-sacm-coswid]

A "COSWID Tag Entry" MUST conform to the following requirements:

- The value of the "type" attribute of the atom:content element MUST be "application/swid+cbor".
- There MUST be one "rolie:property" with the "name" attribute equal to "urn:ietf:params:rolie:property:content-id" and the "value" attribute exactly equal to the decoded "tag-id" element in the attached COSWID Tag (mapped to integer 0). This allows ROLIE consumers to more easily search for COSWID tags without needing to download the tag itself.
- There MUST be one "rolie:property" with the "name" attribute equal to "urn:ietf:params:rolie:property:swd:swname", and the "value" attribute equal to the decoded value of the "swid-name" element in the attached COSWID Tag (mapped to the integer 1). As above, this helps ROLIE consumers search and filter Entries.
- There MAY be a property element with the "name" attribute equal to "urn:ietf:params:rolie:property:swd:swversion". When this property appears, it’s value MUST be equal to the decoded value of the tag-version element in the attached COSWID Tag (mapped to the integer 12).

7. atom:link Extensions

This section defines additional link relationships that implementations MUST support. These relationships are not registered in the Link Relation IANA table as their use case is too narrow. Each relationship is named and described.

Name	Description
ancestor	Links to a software descriptor resource that defines an ancestor of the software being described by this Entry. This is usually a previous version of the software.
descendent	Links to a software descriptor resource that defines an descendent of the software being described by this Entry. This is usually a more recent version or edition of the software.
patches	Links to a software descriptor resource that defines the software being patched by this software
patchedby	Links to a software descriptor resource that defines the patch or update itself that can be or has been applied to this software.
requires	Links to a software descriptor resource that defines a piece of software required for this software to function properly, i.e., a dependency.
requiredBy	Links to a software descriptor resource that defines a piece of software that requires this software to function properly.
installs	Links to a software descriptor resource that defines the software that is installed by this software.
installedBy	Links to a software descriptor resource that defines the software package that installs this software.

patchesVulnerability	Links to a vulnerability that this software update fixes. Used for software descriptors that describe software patches or updates.
hasVulnerability	Links to a vulnerability description object that details a vulnerability that this software has.

Table 1: Link Relations for Resource-Oriented Lightweight Indicator Exchange

8. IANA Considerations

8.1. software-descriptor information-type

IANA has added an entry to the "ROLIE Security Resource Information Type Sub-Registry" registry located at <https://www.iana.org/assignments/rolie/category/information-type> .

The entry is as follows:

name: software-descriptor

index: TBD

reference: This document, Section 4

8.2. swd:swname property

IANA has added an entry to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

The entry is as follows:

name: property:swd:swname

Extension IRI: urn:ietf:params:rolie:property:swd:swname

Reference: This document, Section 5.1

Subregistry: None

8.3. swd:swversion property

IANA has added an entry to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

The entry is as follows:

name: property:swd:swversion

Extension IRI: urn:ietf:params:rolie:property:swd:swversion

Reference: This document, Section 5.1

Subregistry: None

8.4. swd:swcreator property

IANA has added an entry to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

The entry is as follows:

name: property:swd:swcreator

Extension IRI: urn:ietf:params:rolie:property:swd:swcreator

Reference: This document, Section 5.1

Subregistry: None

9. Security Considerations

Use of this extension implies dealing with the security implications of both ROLIE and of software descriptors in general. As with any data, care should be taken to verify the trustworthiness and veracity of the descriptor information to the fullest extent possible.

Ideally, software descriptors should be signed by the software manufacturer, or signed by whichever agent processed the source code. Software descriptor documents from these sources are more likely to be accurate than those generated by scraping installed software.

These "authoritative" sources of software descriptor content should consider additional security for their ROLIE repository beyond the typical recommendations, as the central importance of the repository is likely to make it a target.

Version information is often represented differently across manufacturers and even across product releases. If using software version information for low fault tolerance comparisons and searches, care should be taken that the correct version scheme is being used.

10. Normative References

[I-D.ietf-sacm-coswid]
Birkholz, H., Fitzgerald-McKay, J., Schmidt, C., and D. Waltermire, "Concise Software Identification Tags", draft-ietf-sacm-coswid-11 (work in progress), June 2019.

[NISTIR8060]
Waltermire, D., Cheikes, B., Feldman, L., and G. Witte, "Guidelines for the Creation of Interoperable Software Identification (SWID) Tags", NISTIR 8060, April 2016, <https://doi.org/10.6028/NIST.IR.8060>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <https://www.rfc-editor.org/info/rfc4949>.

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, DOI 10.17487/RFC5070, December 2007, <https://www.rfc-editor.org/info/rfc5070>.

[RFC8322] Field, J., Banghart, S., and D. Waltermire, "Resource-Oriented Lightweight Information Exchange (ROLIE)", RFC 8322, DOI 10.17487/RFC8322, February 2018, <https://www.rfc-editor.org/info/rfc8322>.

[SWID] "Information technology - Software asset management - Part 2: Software identification tag", ISO/IEC 19770-2:2015, October 2015, <https://www.iso.org/standard/65666.html>.

Appendix A. Schema

This document does not require any schema extensions.

Appendix B. Examples of Use

Use of this extension in a ROLIE repository will not typically change that repository's operation. As such, the general examples provided by the ROLIE core document would serve as examples. Provided below is a sample software descriptor ROLIE entry:

```
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom"
  xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0">
  <id>dd786dba-88e6-440b-9158-b8fae67ef67c</id>
  <title>Sample Software Descriptor</title>
  <published>2015-08-04T18:13:51.0Z</published>
  <updated>2015-08-05T18:13:51.0Z</updated>
  <summary>A descriptor for a piece of software published by this
  organization. </summary>
  <link rel="self" href="http://www.example.org/rolie/SWD/123456"/>
  <link rel="feed" href="http://www.example.org/rolie/SWD/"/>
  <link rel="requires" href="http://www.example.org/rolie/SWD/78430"/>
  <rolie:property name=urn:ietf:params:rolie:property:swd:swname
      value="Example Software Name"/>
  <category
      scheme="urn:ietf:params:rolie:category:information-type"
      term="software-descriptor"/>
  <rolie:format
      ns="http://standards.iso.org/iso/19770/-2/2015/schema.xsd"/>
  <content type="application/xml"
      src="http://www.example.org/rolie/SWD/123456/data"/>
</entry>
```

Authors' Addresses

Stephen Banghart
NIST
100 Bureau Drive
Gaithersburg, Maryland 20877
USA

Email: stephen.banghart@nist.gov

David Waltermire
NIST
100 Bureau Drive
Gaithersburg, Maryland 20877
USA

Email: david.waltermire@nist.gov

Security Automation and Continuous Monitoring (SACM)
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2019

Q. Lin
L. Xia
Huawei
H. Birkholz
Fraunhofer SIT

October 22, 2018

The Data Model of Network Infrastructure Device Management Plane Security Baseline
draft-lin-sacm-nid-mp-security-baseline-04

Abstract

This document provides security baseline for network device management plane, which is represented by YANG data model. The corresponding configuration values and status values of the YANG data model can be transported between Security Automation and Continuous Monitoring (SACM) components and used for network device security posture assessment.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

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 April 25, 2019.

Table of Contents

1. Introduction

Besides user devices and servers, network devices such as routers, switches, and firewalls are crucial to enterprise network security. The security baseline defined in this document refers to a minimal set of security controls that are essential to provide network security. Organizations can define additional security controls based on the security baseline. Then the security posture of network

devices can be assessed by comparing the configuration values and status values with the required security controls.

Network devices typically perform three planes of operation: management plane, control plane and data plane. All the planes should be protected and monitored. This document focuses on security baseline for management plane. Management plane provides configuration and monitoring services to network administrators or device owners. Unauthorized access, insecure access channels, weak cryptographic algorithms are common security issues that break management plane security. A number of security best practices have been proposed to deal with these security issues, such as disabling unused services and ports, discarding insecure access channels, and enforcing strong user authentication and authorization. In this document, we provide a minimal set of security controls that are expected to be widely applicable to common network devices. To assess security posture of network devices, the configurations that are effective on network devices and the current status of the networks devices will be compared with the reference values defined by an organization or a third party.

YANG data model is used to describe the security baseline defined in this document. [I-D.birkholz-sacm-yang-content] defines a method to construct the YANG data model scheme for network device security posture assessment by brokering YANG push telemetry through SACM statements. In this document, we follow the same way to define the YANG output for network device security posture based on the [I-D.ietf-sacm-information-model].

Besides management plane, the security baselines for control plane, data plane, and infrastructure layer of network infrastructure devices are described in [I-D.dong-sacm-nid-cp-security-baseline], [I-D.xia-sacm-nid-dp-security-baseline] and [I-D.dong-sacm-nid-infra-security-baseline] respectively.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Terminology

This document uses the terms defined in [RFC7950] and [RFC8342].

4. Tree Diagrams

Tree diagram defined in [RFC8340] is used to represent the YANG data model of network device management plane security. The meaning of the symbols used in the tree diagram and the syntax are as follows:

- A module is identified by "module:" followed the module-name. The top-level data nodes defined in the module, offset by 2 spaces. Submodules are represented in the same fashion as modules, but are identified by "submodule:" followed the (sub)module-name.

- Groupings, offset by 2 spaces, and identified by the keyword "grouping" followed by the name of the grouping and a colon (":") character.

- Each node in the tree is prefaces with "+--". Schema nodes that are children of another node are offset from the parent by 3 spaces.

- Brackets "[" and "]" enclose list keys.

- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" means state data (read-only), and "-u" indicates the use of a predefined grouping.

- Symbols after data node names: "?" means an optional leaf, choice, anydata, or anyxml, "!" means a presence container, and "*" denotes a "list" or "leaf-list".

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- At times when the composition of the nodes within a module schema is not important in the context of the presented tree, sibling nodes and their children can be collapsed using the notation "..." in place of the text lines used to represent the summarized nodes.

- Curly brackets and a question mark "{...}?" are combined to represent the features that node depends on.

5. Data Model Structure

The security baseline defined in this document consists of security configuration and runtime security status for administration, system management, port management, log, files.

- Administration security

- System management security
- Port management security
- Log security
- File security

A multitude of YANG modules for network devices and network protocols have been defined in IETF. Several RFCs and drafts model some parts of management plane security. But an overall data model of management plane security is still missing. New modules, groupings, and nodes are defined in this document as supplements. And the existing YANG modules are reused. Appendix A provides a summary of existing YANG modules and the relationship to the security baseline defined in this document.

5.1. Administration Security

5.1.1. Administrative Account Security

In order to provide administrative accounts, security controls on account properties and passwords should be applied. The commonly applied security controls include limiting the length of account name, checking the password complied to the complexity policy, forbidding the use of some strings in password, blocking accounts after several login fails, etc. The following data model illustrates these kinds of security controls.

```
module: ietf-admin-account-security
  +--rw ietf-admin-account-security
     +--rw account-security-policy {account-security}?
     |  +--rw policy-status?          boolean
     |  +--rw account-aging-period?   uint64
     |  +--rw account-name-minlen?    uint64
     +--rw pwd-security-policy {pwd-security}?
     |  +--rw expire-days?           uint64
     |  +--rw prompt-days?           uint64
     |  +--rw change-check?          boolean
     |  +--rw complexity-check?      boolean
     |  +--ro history-pwd-num?       uint64
     |  +--rw pwd-minlen?            uint64
     |  +--rw forbidden-word-rules?
     |     +--rw forbidden-word-rule* [forbidden-word]
     |        +--rw forbidden-word   string
     +--rw login-failed-limit {login-failed-block}?
        +--rw failed-times?          uint64
        +--rw period?                uint64
        +--rw reactive-time?         uint64
```

5.1.2. Administrator Access Security

Network devices typically can be managed through command line interface (CLI) or web user interface. Insecure access channels (e.g., Telnet), can expose the devices to threats and attacks. Therefore, SSH-based access channels and HTTPS-based web channels should be used. Besides, the right version of the protocols should be chosen. For example, SSHv1 is considered not secure, SSHv2 is recommended. And draft [I-D.ietf-tls-oldversions-deprecate] will formally deprecates Transport Layer Security (TLS) versions 1.0 [RFC2246] and 1.1 [RFC4346] and moves these documents to the historic state.

```
module: ietf-admin-access-security
  +--rw ietf-admin-access-security
     +--rw console
     |  +--rw auth-mode?               auth-mode-type
     |  +--rw privilege-level?         uint8
     +--rw vtys
     |  +--rw vty* [vty-number]
     |     +--rw vty-number            uint8
     |     +--rw auth-mode             auth-mode-type
     |     +--rw privilege-level       uint8
     |     +--rw acl-name-list*        string
     |     +--rw ip-block-enable       boolean
     |     +--rw ip-block-limit {ip-block-config}?
     |        +--rw failed-times?      uint64
     |        +--rw period?            uint64
     |        +--rw reactive-time?     uint64
     +--rw ssh
     |  +--rw ssh-enable?              boolean
     |  +---u ssh-server-attribute-grouping
     |  +---u ssh-security-harden-grouping
     |  +--rw ip-block-enable          boolean
     |  +--rw ip-block-limit {ip-block-config}?
     |     +--rw failed-times?         uint64
     |     +--rw period?               uint64
     |     +--rw reactive-time?        uint64
     +--rw web  {web-interface}?
        +--rw privilege-level?         uint8
        +--rw http-server-interface?   string
        +--rw https-ipv4-enable?       boolean
        +--rw https-ipv6-enable?       boolean
        +--rw https-source-port?       inet:port-number
        +--rw https-timeout?           uint32
        +--rw acl-name-list*?          string
        +--rw ip-block-enable          boolean
        +--rw ip-block-limit {ip-block-config}?
        |  +--rw failed-times?         uint64
        |  +--rw period?               uint64
        |  +--rw reactive-time?        uint64
        +---u tls-server-attribute-grouping
```

[I-D.ietf-netconf-ssh-client-server] defines "ssh-server-grouping" for configuring SSH server and does not consider the underlying transport parameters. And it reuses the groupings defined in [I-D.ietf-netconf-keystore]. Because this document focuses on the security configurations that are actively in use when the network device acts as a SSH server, the "ssh-server-attribute-grouping" defined here tailors the "private-key" node and the "certificate-

expiration" notification of "ssh-server-grouping". The tree diagram of grouping "ssh-server-attribute-grouping":

```
  grouping ssh-server-attribute-grouping:
    +--rw server-identity
    |  +--rw host-key* [name]
    |     +--rw name   string
    |     +--rw (host-key-type)
    |        +--:(public-key)
    |        |  +--rw (local-or-keystore)
    |        |     +--:(local)
    |        |     |  +---u ks:public-key-grouping
    |        |     +--:(keystore) {ks:keystore-implemented}?
    |        |        +--rw ref?  ks:asymmetric-key-certificate-ref
    |        +--:(certificate) {sshcmn:ssh-x509-certs}?
    |           +--rw (local-or-keystore)
    |              +--:(local)
    |              |  +---u ks:public-key-grouping
    |              |  +---u ks:trust-anchor-cert-grouping
    |              +--:(keystore) {ks:keystore-implemented}?
    |                 +--rw ref?  ks:asymmetric-key-certificate-ref
    +--rw client-cert-auth {sshcmn:ssh-x509-certs}?
    |  +--rw pinned-ca-certs?       ta:pinned-certificates-ref
    |  +--rw pinned-client-certs?   ta:pinned-certificates-ref
    +--rw transport-params {ssh-server-transport-params-config}?
       +---u sshcmn:transport-params-grouping
```

Besides the security configurations defined "ssh-server-attribute-grouping", there are several other features related the secure use and configuration of SSH, such as which SSH version is used, whether the network device support to be compatible with earlier SSH versions, whether the port number has been changed, etc. The "ssh-security-harden-grouping" includes these kind of security configurations and state. The tree diagram of grouping "ssh-security-harden-grouping":

```
  grouping ssh-security-harden-grouping:
    +--ro ssh-version                uint32
    +--rw ssh-server-port?           inet:port-number
    +--rw ssh-rekey-interval?        uint32
    +--rw ssh-timeout?               uint32
    +--rw ssh-retry-times?           uint32
    +--rw ssh1x-compatible?          boolean
    +--rw ssh-server-interface?      string
```

[I-D.ietf-netconf-tls-client-server] defines "tls-server-grouping" for configuring TLS server and does not consider the underlying transport parameters. And it reuses the groupings defined in

[I-D.ietf-netconf-keystore]. Because this document focuses on the security configurations that are actively in use when the network device acts as a web server and build connections through HTTPS, the "tls-server-attribute-grouping" defined here tailors the "private-key" node and the "certificate-expiration" notification of "tls-server-grouping". The tree diagram of grouping "tls-server-attribute-grouping":

```
  grouping tls-server-attribute-security-grouping:
    +--rw server-identity
    |  +--rw (local-or-keystore)
    |     +--:(local)
    |     |  +---u ks:public-key-grouping
    |     |  +---u ks:trust-anchor-cert-grouping
    |     +--:(keystore) {ks:keystore-implemented}?
    |        +--rw ref?   ks:asymmetric-key-certificate-ref
    +--rw client-auth
    |  +--rw pinned-ca-certs?       ta:pinned-certificates-ref
    |  +--rw pinned-client-certs?   ta:pinned-certificates-ref
    +--rw hello-params {tls-server-hello-params-config}?
       +--rw tls-versions
       |  +--rw tls-version*     identityref
       +--rw cipher-suites
          +--rw cipher-suite*    identityref
```

5.1.3. AAA

Authentication, Authorization, and Accounting (AAA) provides user management for network devices. RADIUS (Remote Authentication Dial In User Service) and TACACS+ (Terminal Access Controller Access Control System) are the commonly used AAA mechanisms. In order to implement AAA, network devices act as AAA clients to communicate with AAA servers. [RFC7317] defined YANG module for client to configure the RADIUS authentication server information. In this document, authentication, authorization and accounting schemes, as well as AAA server lists are all included.

```
module: ietf-aaa-security
  +--rw ietf-aaa-security
     +--rw authentication-scheme* [authen-scheme-name]
     |  +--rw authen-scheme-name   string
     |  +--rw authen-mode*         aaa-authen-mode
     |  +--rw authen-type?         radius-authen-type
     |  +--rw authen-fail-policy?  boolean
     +--rw authorization-scheme* [author-scheme-name]
     |  +--rw author-scheme-name   string
     |  +--rw author-mode*         aaa-author-mode
     |  +--rw cmd-author-mode*     aaa-cmd-author-mode
     +--rw accounting-scheme* [account-scheme-name]
     |  +--rw account-scheme-name  string
     |  +--rw account-mode?        aaa-account-name
     +--rw radius-security
     |  +--rw radius-authen-servers* [address]
     |  |  +--rw address         inet:host
     |  |  +--rw port?           inet:port-number
     |  +--rw radius-author-servers*? [address]
     |  |  +--rw address         inet:host
     |  |  +--rw port?           inet:port-number
     |  +--rw radius-account-servers* [address]
     |     +--rw address         inet:host
     |     +--rw port?           inet:port-number
     +--rw tacacs-security {tacacs-supported}?
        +--rw tacacs-authen-servers* [address]
        |  +--rw address         inet:host
        |  +--rw port?           inet:port-number
        +--rw tacacs-author-servers*? [address]
        |  +--rw address         inet:host
        |  +--rw port?           inet:port-number
        +--rw tacacs-account-servers* [address]
           +--rw address         inet:host
           +--rw port?           inet:port-number
```

5.1.4. Administrator Access Statistics

The statistics of the current online administrators, the failed login attempts and the blocked addresses are useful for the monitoring of network infrastructure devices.

```
module: ietf-admin-access-statistics
  +--ro ietf-admin-access-statistics
     +--ro online
     |  +--ro total-online-users      uint32
     |  +--ro online-admin-list  {display-online-info}?
     |     +--ro online-users* [account-name]
     |        +--ro account-name      string
     |        +--ro ip-address        inet:ip-address-no-zone
     |        +--ro mac-address       yang:mac-address
     +--ro ip-block-list
        +--ro blocked-ip* [ip-address]
           +--ro ip-address           inet:ip-address-no-zone
           +--ro vpn-instance         string
           +--ro state                ip-block-state-type
           +--ro authen-fail-account  uint32
```

5.2. System Management Security

5.2.1. SNMP Management Security

Simple Network Management Protocol (SNMP) is a network management standard to monitor network devices. Three SNMP versions are available: SNMPv1, SNMPv2c, and SNMPv3. [RFC7407] defines community-based security model for SNMPv1 and SNMPv2c, view-based access control model and user-based security model, transport security model for SNMPv3. SNMPv1 and SNMPv2c are lack of authentication and message encryption, which could facilitate unauthorized access to network devices. SNMPv3 needs to be used to authenticate and encrypt payloads. The "ietf-snmp-security" module defined in this section reuses the definitions in [RFC7407], but some modifications and eliminations are made. As this module only focuses on security controls and status of SNMP, the detailed transport information such as IP address and port are not included, while the transport protocol used is under consideration. And the subtree for key configuration is also not needed for user-based security model, but the authentication protocol or encryption protocol used is included.

```
module: ietf-snmp-security
  +--rw ietf-snmp-security
     +--rw snmp-enable?       boolean
     +--rw engine
     |  +--rw enabled?        boolean
     |  +--rw listen* [name]
     |  |  +--rw name         snmp:identifier
     |  |  +--rw transport    snmp-transport-type
     |  +--rw version         snmp-version-type
     |  +--rw enable-authen-traps?  boolean
     +--rw target* [name]
```

```
      |  +--rw name       snmp:identifier
      |  +--rw transport  snmp-transport-type
      |  +--rw target-params   snmp:identifier
      +--rw target-params* [name]
      |  +--rw name        snmp:identifier
      |  +--rw (params)?
      |     +--:(usm)
      |     |  +---u snmp:usm-target-params
      |     +--:(tsm) {snmp:tsm}?
      |     |  +---u snmp:tsm-target-params
      +--rw vacm
      |  +--ro vacm-enable?     boolean
      |  +--rw group* [name]
      |  |  +--rw name                snmp:group-name
      |  |  +--rw member*  [security-name]
      |  |  |  +--rw security-name    snmp:security-name
      |  |  |  +--rw security-model*  snmp:security-model
      |  |  +--rw access* [context security-model security-level]
      |  |     +--rw context          snmp:context-name
      |  |     +--rw context-match?   enumeration
      |  |     +--rw security-model   snmp:security-model-or-any
      |  |     +--rw security-level   snmp:security-level
      |  |     +--rw read-view?       snmp:view-name
      |  |     +--rw write-view?      snmp:view-name
      |  |     +--rw notify-view?     snmp:view-name
      |  +--rw view* [name]
      |     +--rw name      vacm:view-name
      |     +--rw include*  snmp:wildcard-object-identifier
      |     +--rw exclude*  snmp:wildcard-object-identifier
      +--rw usm
      |  +--ro usm-enable?    boolean
      |  +--rw local
      |  |  +---u user-auth-priv
      |  +--rw remote
      |     +---u user-auth-priv
      +--rw tsm   {tsm}?
         +--ro tsm-enable?    boolean
```

The tree diagram of grouping "user-auth-priv":

```
  grouping user-auth-priv:
    +--rw user* [name]
           +--rw name              snmp:identifier
       +--rw auth-protocol     auth-pro-type
       +--rw priv-protocol     priv-pro-type
```

5.2.2. NETCONF Management Security

The NETCONF server model defined in [I-D.ietf-netconf-netconf-client-server] supports both the SSH and TLS transport protocols. The "ietf-netconf-security" module defined in this section only reused the security related subtrees and replaces the SSH and TLS related groupings with those defined in "ietf-admin-access-security" module.

```
module: ietf-netconf-security
  +--rw ietf-netconf-security
     +--rw netconf-enable?     boolean
     +--rw listen {ncs:listen}?
     |  +--rw endpoint* [name]
     |     +--rw name          string
     |     +--rw (transport)
     |        +--:(ssh) {ssh-listen}?
     |        |  +--rw port    inet:port-number
     |        |  +---u accsec:ssh-server-attribute-grouping
     |        +--:(tls) {tls-listen}?
     |           +--rw port    inet:port-number
     |           +---u accsec:tls-server-attribute-grouping
     +--rw call-home {call-home}?
        +--rw netconf-client* [name]
           +--rw name         string
           +--rw endpoints
              +--rw endpoint* [name]
                 +--rw name   string
                 +--rw (transport)
                    +--:(ssh) {ssh-call-home}?
                    |  +--rw port    inet:port-number
                    |  +---u accsec:ssh-server-attribute-grouping
                    +--:(tls) {tls-call-home}?
                       +--rw port    inet:port-number
                       +---u accsec:tls-server-attribute-grouping
```

5.3. Port Management Security

As it is suggested to disable unused service and ports, the current status (open or shut-down) of the ports that are available on the network devices can be retrieved and compared with the communication matrix to check the device security posture.

```
module: ietf-port-management-security
  +--rw ietf-port-management-security
     +--rw port-list*  [port-number]
        +--rw port-number       inet:port-number
        +--rw port-status        boolean
```

5.4. Log Security

To monitor the running status and diagnose faults or attacks on network devices, the activities of network administrators, the operations conducted on devices, and the security notification of abnormal events need to be recorded. Besides, policy should be defined to deal with log overflow. Log records can be outputted to console, or stored locally, or outputted to remote Syslog server. The following defined "ietf-log-security" module reuses the security configuration of log remote transfer in [I-D.ietf-netmod-syslog-model], and adds access control for locally stored log files.

```
module: ietf-log-security
  +--rw ietf-log-security
     +--rw alert-notification
     |  +--rw login-fail-threshold         uint8
     |  +--rw system-abnormal              boolean
     |  +--rw attack                       boolean
     |  +--rw log-overflow-lost            boolean
     +--rw (log-overflow-action)
     |  +--:(rewrite-when-overflow)        boolean
     |  |  +--ro rewrite-numbers           uint16
     |  +--:(discard-new-logs)             boolean
     |     +--ro discard-numbers           uint16
     +--rw (log-mode)
        +--:(file) {file-action}?
        |  +--rw user-level-for-read       uint8
        |  +--rw user-level-for-delete     uint8
        +--:(remote) {remote-action}?      [I-D.ietf-netmod-syslog-model]
           +--rw destination* [name]
              +--rw name                   string
              +--rw (transport)
              |  ...
              +--rw signing! {signed-messages}?
                 ...
```

5.5. File Security

Patches, packages, configuration files, password files are critical system files for network infrastructure devices. Only administrators with certain security privilege levels are allowed to access or operate on these files. For file transfer security, secure protocol should be used.

```
   module: ietf-file-security
     +--rw ietf-file-security
        +--rw role-based-access-control   boolean
        +--rw transport-protocol          file-pro-type
        +--rw (transport)
        |  +--:(sftp) {sftp}?
        |  |  +--rw sftp-enable              boolean
        |  |  +--rw sftp-server-port         inet:port-number
        |  |  +---u accsec:ssh-server-attribute-grouping
        |  |  +---u accsec:ssh-security-harden-grouping
        |  +--:(scp) {scp}?
        |  |  +--rw scp-enable               boolean
        |  |  +--rw scp-server-port          inet:port-number
        |  |  +---u accsec:ssh-server-attribute-grouping
        |  |  +---u accsec:ssh-security-harden-grouping
        |  +--:(ftps) {ftps}?
        |  |  +--rw ftps-enable              boolean
        |  |  +--rw ftps-server-port         inet:port-number
        |  |  +---u accsec:tls-server-attribute-grouping
        |  +--rw ip-block-enable             boolean
        |  +--rw ip-block-limit {ip-block-config}?
        |     +--rw failed-times             uint64
        |     +--rw period                   uint64
        |     +--rw reactive-time            uint64
```

6. Network Infrastructure Device Security Baseline Yang Module

6.1. Module 'ietf-admin-account-security'

```
<CODE BEGINS> file "ietf-admin-account-security@2018-10-16.yang"
module ietf-admin-account-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-admin-account-security";
  prefix acsec;

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";
```

```
  description
    "This YANG module defines ietf-admin-account-security YANG module, which con
tains configurations that are actively in use for account security control, pass
word security control and administrative account block.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  /*
  * features
  */
  feature account-security {
    description
      "If the network device supports this feature, then several security contro
ls on administrative accounts can be conducted.";
  }

  feature pwd-security {
    description
      "If the network device supports this feature, then several security contro
ls on password can be conducted.";
  }

  feature login-failed-block {
    description
      "If the network device supports this feature, an adminstrative account wil
l be blocked for a certain time range when this account login failed several tim
es in a certain period.";
  }

  /*
  * containers
  */

  container account-security-policy {
    if-feature account-security;
    leaf policy-status {
      type boolean;
      description
        "The status of account security policy: enabled, or disabled.";
    }
    leaf account-aging-period {
      type uint64;
      description
        "The aging period of an administrative account.";
    }
    leaf account-name-minlen {
      type uint64;
      description
        "The minimum length of an administrative account name.";
    }
```

```
    description
      "If the network device supports some security controls on administrative a
ccounts, the configuration that is actively in use will be collected.";
  }

  container pwd-security-policy {
    if-feature pwd-security;
    leaf expire-days {
      type uint64;
      description
        "The password validity period.";
    }
    leaf prompt-days {
      type uint64;
      description
        "The period for warning before the password expires.";
    }
    leaf change-check {
      type boolean;
      description
        "Whether it is mandatory to change the password when logining for the fi
rst time: enabled, or disabled.";
    }
    leaf complexity-check {
      type boolean;
      description
        "The status of password complexity check: enabled, or disabled.";
    }
    leaf history-pwd-num {
      type uint64;
      config false;
      description
        "The newly configured password should not be the same as the several pas
t passwords.";
    }
    leaf pwd-minlen {
      type uint64;
      description
        "The minimum length of a password.";
    }
    container forbidden-word-rules {
      list forbidden-word-rule {
        key "forbidden-word";
        leaf forbidden-word {
          type string;
          description
            "A forbidden word in password.";
        }
        description
          "A list of forbidden words that are not allowed to be used in password
.";
      }
```

```
      description
        "Password blacklist.";
    }
    description
      "If the network device supports some security controls on administrative p
asswords, the configuration that is actively in use will be collected.";
  }

  container login-failed-limit {
    if-feature login-failed-block;
    leaf failed-times {
      type uint64;
      description
        "The failed time in a certain period.";
    }
    leaf peroid {
      type uint64;
      description
        "The certain period in which the failed times are counted.";
    }
    leaf reactive-time {
      type uint64;
      description
        "The reactive time after which the account is not blocked.";
    }
    description
      "If the network device suppor this feature, an account will be blocked for
 a certain time range when it failed to login for several times in a certain per
iod.";
  }
}
<CODE ENDS>
```

6.2. Module 'ietf-admin-access-security'

```
module ietf-admin-access-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-admin-access-security";
  prefix accsec;

  import ietf-inet-types {
    prefix inet;
      reference
        "RFC 6991 - Common YANG Data Types.";
  }

  import ietf-ssh-common {
    prefix sshcmn;
      reference
        "draft-ietf-netconf-ssh-client-server - YANG Groupings for SSH Clients a
nd SSH Servers";
  }
```

```
  import ietf-tls-common {
    prefix tlscmn;
      reference
        "draft-ietf-netconf-tls-client-server - YANG Groupings for TLS Clients a
nd SSH Servers";
  }

  import ietf-keystore {
    prefix ks;
      reference
        "draft-ietf-netconf-keystore - YANG Data Model for a Centralized Keystor
e Mechanism";
  }

  import ietf-trust-anchors {
    prefix ta;
      reference
        "draft-ietf-netconf-trust-anchors - YANG Data Model for Global Trust Anc
hors";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-admin-access-security YANG module, which cont
ains security configurations that are actively in use for different access chann
els.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  /*
  * features
  */
  feature web-interface {
    description
      "If the network device supports web interface for administration, then adm
inistrative account can access this device through web interface.";
  }
```

```
  feature ip-block-config {
    description
      "If the network device supports the configuration of ip block function, th
en it can be configured to block the access from a list of IP addresses.";
  }

  feature ssh-server-transport-params-config {
    description
      "SSH transport layer parameters are configurable on an SSH server.";
  }

  feature tls-server-hello-params-config {
    description
      "TLS hello message parameters are configurable on a TLS server.";
  }

  /*
  * typedefs
  */
  typedef auth-mode-type {
    type enumeration {
      enum "none" {
        description
          "Authentication mode: none.";
      }
      enum "password" {
        description
          "Authentication mode: password.";
      }
      enum "aaa" {
         description
           "Authentication mode: aaa.";
      }
    }
    description
      "The Authentication mode of console and vty interface.";
   }

  /*
  * groupings
  */
  grouping ssh-server-attribute-grouping {
    container server-identity {
      list host-key {
        key "name";
        leaf name {
          type string;
          description
            "The name of the host-key.";
```

```
                }
        choice host-key-type {
          mandatory true;
          case public-key {
            choice local-or-keystore {
              case local {
                uses ks:public-key-grouping;
                description
                  "The public key and the corresponding algorithm.";
                          }
              case keystore {
                if-feature ks:keystore-implemented;
                leaf ref {
                  type ks:asymmetric-key-certificate-ref;
                  description
                    "A reference to a value that exists in the keystore.";
                }
                description
                  "The reference of the key pair that stored in the keystore. ";
              }
              description
                "The key pair is locally stored or can be referenced from the ke
ystore.";
            }
            description
              "The host key type is asymmetric key pair.";
          }
          case certificate {
            if-feature sshcmn:ssh-x509-certs;
            choice local-or-keystore {
              case local {
                uses ks:public-key-grouping;
                uses ks:trust-anchor-cert-grouping;
                description
                  "The certificate and the corresponding public key are stored l
ocally.";
              }
              case keystore {
                if-feature ks:keystore-implemented;
                leaf ref {
                  type ks:asymmetric-key-certificate-ref;
                  description
                    "The certificate is referenced by a value that exists in the
 keystore.";
                }
                description
                  "The reference of the certificate that stored in the keystore.
";
              }
              description
                "The certificate is stored locally or can be referenced from the
 keystore.";
            }
```

```
            description
              "The host key type is certificate.";
          }
          description
            "Two types of host key: asymmetric key pair, certificate.";
        }
        description
          "A list of host keys of the network device";
      }
      description
        "The list of host keys the network device (acts as SSH server) will use
to construct its list of algorithms, when sending its SSH-MSG-KEXINIT message, a
se defined in Section 7.1 of RFC 4253.";
      }
    container client-cert-auth {
      if-feature sshcmn:ssh-x509-certs;
      leaf pinned-ca-certs {
        type ta:pinned-certificates-ref;
        description
          "A reference to a list of certificate authority (CA) certificates used
 by the SSH server to authenticate SSH client certificates.";
        reference
          "draft-ietf-netconf-trust-anchors: YANG Data Model for Global Trust An
chors";
      }
      leaf pinned-client-certs {
        type ta:pinned-certificates-ref;
        description
          "A reference to a list of client certificates used by the SSH server t
o authenticate SSH client certificates.";
        reference
          "draft-ietf-netconf-trust-anchors: YANG Data Model for Global Trust An
chors";
      }
      description
        "A reference to a list of pinned certificate authority (CA) certificates
 and a reference to a list of pinned client certificates.";
    }
    container transport-params {
      if-feature ssh-server-transport-params-config;
      uses sshcmn:transport-params-grouping;
      description
        "Configurable parameters of the SSH transport layer.";
    }
    description
      "A reusable grouping of configurations that are actively in use for networ
k devices which act as SSH servers.";
  }

  grouping ssh-security-harden-grouping {
    leaf ssh-version {
      type uint32;
      config false;
      mandatory true;
      description
        "The SSH version that the network device supports.";
```

```
    }
    leaf ssh-server-port {
      type inet:port-number;
      description
        "The port number of SSH server.";
    }
    leaf ssh-rekey-interval {
      type uint32;
      description
        "The interval for updating the key pair of the SSH server.";
    }
    leaf ssh-timeout {
      type uint32;
      description
        "The authentication timeout period of SSH.";
    }
    leaf ssh-retry-times {
      type uint32;
      description
        "The authentication retry times.";
    }
    leaf ssh1x-compatible {
      type boolean;
      description
        "The status of version-compatible function on the SSH server: enabled, d
isabled.";
    }
    leaf ssh-server-interface {
      type string;
      description
        "The source interface of SSH server.";
    }
        description
      "A set of SSH configuration status to enhance security.";
  }

  grouping tls-server-attribute-grouping {
    container server-identity {
      choice local-or-keystore {
          case local {
            uses ks:public-key-grouping;
            uses ks:trust-anchor-cert-grouping;
            description
              "The certificate and the corresponding public key are stored local
ly.";
          }
          case keystore {
            if-feature ks:keystore-implemented;
            leaf ref {
              type ks:asymmetric-key-certificate-ref;
```

```
              description
                "The certificate is referenced by a value that exists in the key
store.";
            }
            description
              "The reference of the certificate that stored in the keystore.";
          }
          description
            "The certificate is stored locally or can be referenced from the key
store.";
      }
      description
        "A locally-defined or referenced end-entity certificate, including any c
onfigured intermediate certificates, the TLS server will present when establishi
ng a TLS connection in its Certificate message, as defined in Section 7.4.2 in R
FC5246.";
        }
    container client-auth {
      leaf pinned-ca-certs {
        type ta:pinned-certificates-ref;
        description
          "A reference to a list of certificate authority (CA) certificates used
 by the TLS server to authenticate TLS client certificates.";
        reference
          "draft-ietf-netconf-trust-anchors: YANG Data Model for Global Trust An
chors";
      }
      leaf pinned-client-certs {
        type ta:pinned-certificates-ref;
        description
          "A reference to a list of client certificates used by the TLS server t
o authenticate TLS client certificates.";
        reference
          "draft-ietf-netconf-trust-anchors: YANG Data Model for Global Trust An
chors";
      }
      description
        "A reference to a list of pinned certificate authority (CA) certificates
 and a reference to a list of pinned client certificates.";
    }
    container hello-params {
      if-feature tls-server-hello-params-config;
      uses tlscmn:hello-params-grouping;
      description
        "Configurable parameters for the TLS hello message.";
    }
    description
      "A reusable grouping of configurations that are actively in use for networ
k devices which act as TLS servers.";
  }


  /*
  * containers
  */
  container console {
    leaf auth-mode {
      type auth-mode-type;
      description
```

```
        "The authentication mode used when administrative accounts login through
 console interface: none, password, AAA.";
    }
    leaf privilege-level {
      type uint8;
      description
        "User privilege level.";
    }
    description
      "Security configurations that are actively in use for console interface.";
  }

  container vtys {
    list vty {
      key "vty-number";
      leaf vty-number {
        type uint8;
        description
          "The number of the vty interface.";
      }
      leaf auth-mode {
        type auth-mode-type;
        mandatory true;
        description
          "The authentication mode used when administrator login through vty int
erface: none, password, AAA.";
      }
      leaf privilege-level {
        type uint8;
        mandatory true;
        description
          "User privilege level.";
      }
      leaf-list acl-name-list {
        type string;
        description
          "The name of the acl.";
      }
      leaf ip-block-enable {
        type boolean;
        mandatory true;
        description
          "The status of ip block function: enabled, or disabled.";
      }
      container ip-block-limit {
        if-feature ip-block-config;
                leaf failed-times {
          type uint64;
                  description
            "The failed times in a certain perid.";
```

```
        }
        leaf peroid {
          type uint64;
                  description
            "The certain period in which the failed times are counted.";
        }
        leaf reactive-time {
          type uint64;
          description
            "The reactive time after which the address is not blocked.";
        }
        description
          "If the login from an address failed several times in a certain period
, this address will be blocked for a certain time range.";
      }
      description
        "Security configurations that are actively in use for a vty interface.";
    }
    description
      "A list of security configurations that are actively in use for each vty i
nterface.";
  }

  container ssh {
    uses ssh-server-attribute-grouping;
    uses ssh-security-harden-grouping;
    leaf ssh-enable {
      type boolean;
      description
        "The status of SSH server: enabled, or disabled.";
    }
    leaf ip-block-enable {
      type boolean;
      description
        "The status of ip block function: enabled, or disabled.";
    }
    container ip-block-limit {
      if-feature ip-block-config;
      leaf failed-times {
        type uint64;
                description
          "The failed times in a certain perid.";
      }
      leaf peroid {
        type uint64;
                description
          "The certain period in which the failed times are counted.";
      }
      leaf reactive-time {
        type uint64;
```

```
                description
            "The reactive time after which the address is not blocked.";
        }
        description
          "If the login from an address failed several times in a certain period,
this address will be blocked for a certain time range.";
      }
      description
        "Security configurations that are actively in use for SSH-based access cha
nnel.";
    }

    container web {
      if-feature web-interface;
      uses tls-server-attribute-grouping;
      leaf auth-mode {
        type auth-mode-type;
        description
          "The authentication mode used when administrator login through web inter
face: none, password, AAA.";
      }
      leaf privilege-level {
        type uint8;
        description
          "User privilege level.";
      }
      leaf http-server-interface {
        type string;
        description
          "The source interface of web server.";
      }
      leaf https-ipv4-enable {
        type boolean;
        description
          "The status of ipv4 https server: enabled, disabled.";
      }
      leaf https-ipv6-enable {
        type boolean;
        description
          "The status of ipv6 https server: enabled, disabled.";
      }
      leaf https-source-port {
       type inet:port-number;
       description
         "The port number of web server.";
      }
      leaf https-timeout {
        type uint32;
        description
          "The authentication timeout period of https.";
      }
```

```
    leaf ip-block-enable {
      type boolean;
      description
        "The status of ip block function: enabled, or disabled.";
    }
    container ip-block-limit {
      if-feature ip-block-config;
      leaf failed-times {
        type uint64;
        description
          "The failed times in a certain perid.";
      }
      leaf peroid {
        type uint64;
        description
          "The certain period in which the failed times are counted.";
      }
      leaf reactive-time {
        type uint64;
        description
          "The reactive time after which the address is not blocked.";
      }
      description
        "If the login from an address failed several times in a certain period,
this address will be blocked for a certain time range.";
    }
    description
      "If the network device supports web interface. The configuration status of
 the web server.";
  }
}
```

6.3. Module 'ietf-aaa-security'

```
<CODE BEGINS> file "ietf-aaa-security@2018-10-16.yang"
module ietf-aaa-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-aaa-security";
  prefix aaasec;

  import ietf-inet-types {
    prefix inet;
      reference
        "RFC 6991 - Common YANG Data Types.";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
```

```
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-aaa-security YANG module, which contains conf
igurations of AAA.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  /*
  * features
  */
  feature tacacs-supported {
    description
      "Whether the device supports TACACS+ based Authentication, Authorization,
and Accounting.";
  }

  /*
  * typedefs
  */
  typedef aaa-authen-mode {
    type enumeration {
      enum "invalid" {
        description
          "Invalid authentication mode.";
      }
      enum "local" {
        description
          "Local authentication mode.";
      }
      enum "tacacs" {
        description
          "TACACS authentication mode. ";
      }
      enum "radius" {
        description
          "RADIUS authentication mode. ";
       }
```

```
      enum "none" {
        description
          "In this mode, users can pass with authentication.";
      }
      enum "radius-proxy" {
        description
          "RADIUS proxy authentication mode.";
      }
    }
    description
      "Diffrent types of authentication modes.";
  }

  typedef radius-authen-type {
    type enumeration {
      enum "pap" {
        description
          "PAP authentication.";
      }
      enum "chap" {
        description
          "CHAP authentication.";
      }
    }
    description
      "Different authentication types of RADIUS authentication.";
  }

  typedef aaa-author-mode {
    type enumeration {
      enum "invalid" {
        description
          "Invalid authorization mode.";
      }
      enum "local" {
        description
          "Local authorization mode.";
      }
      enum "tacacs" {
        description
          "TACACS authorization mode.";
      }
      enum "if-authenticated" {
        description
          "If-authenticated mode: If users pass the authentication and the authe
ntication is not in this mode, it indicates that the user authorization is passe
d. Otherwise, the authorization is not passed.";
      }
      enum "none" {
        description
```

```
          "Users can pass without authorization.";
      }
    }
    description
      "Different types of AAA authorization modes.";
  }

  typedef aaa-cmd-author-mode {
    type enumeration {
      enum "invalid" {
        description
          "Invalid command line authorization mode.";
      }
      enum "local" {
        description
          "Local command line authorization mode.";
      }
      enum "tacacs" {
        description
          "Specifies that the TACACS mode is applied.";
      }
    }
    description
          "Different types of command line authorization modes.";
  }

  typedef aaa-account-mode {
    type enumeration {
      enum "invalid" {
        description
          "invalid accounting mode.";
      }
      enum "radius" {
        description
          "RADIUS accounting mode. ";
      }
      enum "tacacs" {
        description
          "TACACS accounting mode. ";
      }
      enum "none" {
        description
          "In this mode, users do not be accounting.";
      }
    }
    description
      "Different types of accounting modes.";
  }
```

```
  /*
  * lists & containers
  */
  list authentication-scheme {
    key "authen-scheme-name";
    leaf authen-scheme-name {
      type string;
      description
        "The name of the authentication scheme.";
    }
    leaf-list authen-mode {
      type aaa-authen-mode;
      description
        "A list of authentication modes with different preference level. The sec
ond, third, and the following authentication mode is used only when the first au
thentication mode does not respond.";
    }
    leaf authen-type {
      type radius-authen-type;
      description
        "Authentication type of RADIUS: PAP, CHAP.";
    }
    leaf authen-fail-policy {
      type boolean;
      description
        "The policy to be adopted after user authentication fail: force the user
 to be offline, allow user login to a domain with access control.";
    }
    description
      "Authentication scheme list.";
  }

  list authorization-scheme {
    key "author-scheme-name";
    leaf author-scheme-name {
      type string;
      description
        "The name of the authorization scheme.";
    }
    leaf-list auhtor-mode {
      type aaa-author-mode;
      description
        "A list of authorization modes with different preference level. The seco
nd, third, and the following authorization mode is used only when the first auth
orization mode does not respond.";
    }
    leaf-list cmd-auhtor-mode {
      type aaa-cmd-author-mode;
      description
        "A list of command line authorization modes with different preference le
vel. The second, third, and the following command line authorization mode is use
d only when the first command line authorization mode does not respond.";
    }
    description
      "Authorization scheme list.";
```

```
  }

  list accounting-scheme {
    key "account-scheme-name";
    leaf account-scheme-name {
      type string;
      description
        "The name of the accounting scheme.";
    }
    leaf account-mode {
      type aaa-account-mode;
      description
        "Accounting mode.";
    }
    description
      "Accounting scheme list.";
  }

  container radius-security {
    list radius-authen-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the authentication server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the authentication server.";
      }
      description
        "A list of RADIUS authentication servers";
    }
    list radius-author-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the authorization server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the authorization server.";
      }
      description
        "A list of RADIUS authorization servers";
```

```
    }
    list radius-account-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the accounting server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the accounting server.";
      }
      description
        "A list of RADIUS accounting servers";
    }
    description
      "RADIUS authentication servers, authorization servers and accounting serve
rs.";
  }

  container tacacs-security {
        if-feature tacacs-supported;
    list tacacs-authen-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the authentication server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the authentication server.";
      }
      description
        "A list of TACACS+ and TACACS+ compatible authentication servers";
    }
    list tacacs-author-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the authorization server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the authorization server.";
```

```
      }
      description
        "A list of TACACS+ and TACACS+ compatible authorization servers";
    }
    list tacacs-account-servers {
      key "address";
      leaf address {
        type inet:host;
        description
          "The ip address of the accounting server.";
      }
      leaf port {
        type inet:port-number;
        description
          "The port number of the accounting server.";
      }
      description
        "A list of TACACS+ and TACACS+ compatible accounting servers";
    }
    description
      "TACACS+ and TACACS+ compatible authentication servers, authorization serv
ers, and accounting servers.";
  }
}
<CODE ENDS>
```

6.4. Module 'ietf-admin-access-statistics'

```
<CODE BEGINS> file "ietf-admin-access-statistics@2018-10-16.yang"
module ietf-admin-access-statistics {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-admin-access-statistics";
  prefix stat;

  import ietf-inet-types {
    prefix inet;
      reference
        "RFC 6991 - Common YANG Data Types.";
  }

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991 - Common YANG Data Types.";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";
```

```
  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-admin-access-statistics YANG module, which co
ntains online administrator lists, ip addresses authentication failure or blocke
d ip addresses.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  /*
  * features
  */
  feature display-online-info {
    description
      "If the device supports reporting the details of administrative accounts t
hat are currenlty online.";
  }

  /*
  * typedef
  */
  typedef ip-block-state-type {
    type enumeration {
      enum "authenfail" {
        description
          "Authentication fialed State";
      }
      enum "blocked" {
        description
          "BLOCKED State";
      }
    }
    description
      "The status of an login failed IP address.";
  }

  /*
  * containers
```

```
  */
  container online {
    leaf total-online-users {
      type uint32;
      config false;
      description
        "The number of administrators that are current online.";
    }
    container online-admin-list {
      if-feature display-online-info;
      list online-users {
        key "account-name";
        leaf account-name {
          type string;
          description
            "The account name of the online account.";
        }
        leaf ip-address {
          type inet:ip-address-no-zone;
          config false;
                  description
            "The ip address of the online account.";
        }
        leaf mac-address {
          type yang:mac-address;
          config false;
                  description
            "The MAC address of the online account.";
        }
        description
          "Online adminstrator list.";
      }
      description
        "If the device supports providing information of online administrators,
a list of account details are provided.";
    }
    description
      "Online administrator statistics and details.";
  }

  container ip-block-list {
    list blocked-ip {
      key "ip-address";
      leaf ip-address {
        type inet:ip-address-no-zone;
        description
          "The blocked IP address.";
      }
      leaf vpn-instance {
```

```
        type string;
        config false;
        description
          "The VPN instance of the blocked IP address.";
          }
      leaf state {
        type ip-block-state-type;
        config false;
        description
          "The status of an login failed IP address.";
          }
      leaf authen-fail-account {
        type uint32;
        config false;
        description
          "The number of the login failed attempts.";
          }
      description
        "The list of blocked IP addresses and related information.";
    }
    description
      "The information of blocked IP addresses and related information.";
  }
}
<CODE ENDS>
```

6.5. Module 'ietf-snmp-security'

```
<CODE BEGINS> file "ietf-snmp-security@2018-10-16.yang"
module ietf-snmp-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-snmp-security";
  prefix snmpsec;

  import ietf-snmp {
    prefix snmp;
        reference
      "RFC 7407.";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
```

```
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-snmp-security YANG module.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  feature tsm {
    description
      "Whether the network device supports Transport Security Model for SNMP.";
  }

  /*
  * typedef
  */
  typedef snmp-transport-type {
    type enumeration {
      enum "udp" {
        description
          "SNMP over UDP.";
      }
      enum "ssh" {
        description
          "SNMP over SSH.";
      }
      enum "tls" {
        description
           "SNMP over TLS.";
      }
      enum "dtls" {
        description
          "SNMP over DTLS.";
      }
    }
    description
      "The transport channels on which the SNMP engine listens.";
   }

  typedef snmp-version-type {
    type enumeration {
```

```
      enum "v1" {
        description
          "SNMPv1";
      }
      enum "v2c" {
        description
          "SNMPv2c";
      }
      enum "v3" {
        description
           "SNMPv3";
      }
    }
    description
      "The version of SNMP protocol";
  }

  typedef auth-pro-type {
    type enumeration {
      enum "none" {
        description
          "Do not enable the authentication of messages sent on behalf of the us
er.";
      }
      enum "md5" {
        description
          "HMAC-MD5-96 authentication protocol";
      }
      enum "sha" {
        description
          "HMAC-SHA-96 authentication protocol";
      }
    }
    description
      "An indication of whether messages sent on behalf of this user can be auth
enticated, and if so, the type of authentication protocol which is used: MD5, SH
A.";
    reference
      "RFC 3414";
  }

  typedef priv-pro-type {
    type enumeration {
      enum "none" {
        description
          "Do not enable the encryption of messages sent on behalf of the user."
;
      }
      enum "des" {
        description
          "DES is used to encrypt messages sent on behalf of the user.";
      }
```

```
      enum "aes" {
        description
          "AES is used to encrypt messages sent on behalf of the user.";
      }
    }
    description
      "An indication of whether messages sent on behalf of this user can be prot
ected from disclosure, and if so, the type of privacy protocol which is used: ED
S, AES.";
    reference
      "RFC 3414 & RFC 3826";
  }

  /*
  * grouping
  */
  grouping user-auth-priv {
    list user {
        key "name";
        leaf name {
          type snmp:identifier;
          description
            "The identifier that represents a user.";
        }
                leaf auth-protocol {
          type auth-pro-type;
          description
            "The type of authentication protocol: none, md5, sha.";
        }
                leaf priv-protocol {
          type priv-pro-type;
          description
            "The type of encryption protocol: none, des, aes.";
        }
        description
          "A list of users and their corresponding authProtocol, privProtocol.";
      }
      description
        "A grouping that represents a list of users and their corresponding auth
Protocol, privProtocol.";
      reference
        "RFC 3414";
  }

  leaf snmp-enable {
    type boolean;
    description
      "whether SNMP is used.";
  }

  /*
```

```
  * containers
  */
  container engine {
    leaf enabled {
      type boolean;
      description
        "The status of the SNMP engine: enabled, disabled.";
    }
    list listen {
      key "name";
      leaf name {
        type snmp:identifier;
        description
          "The name of a transport channel on which the SNMP engine listens.";
      }
      leaf transport {
        type snmp-transport-type;
        description
          "The transport protocol that SNMP uses.";
      }
      description
        "A list of transport channels on which the SNMP engine listens.";
    }
    leaf version {
      type snmp-version-type;
      description
        "SNMP version used by the SNMP engine.";
    }
    leaf enable-authen-traps {
      type boolean;
      description
        "Whether the SNMP entity is permitted to generate authenticationFailure
traps.";
      reference
        "RFC 3418: Management Information Base (MIB) for the Simple Network Mana
gement Protocol (SNMP) SNMPv2-MIB.snmpEnableAuthenTraps";
    }
    description
      "The security configurations for SNMP engine.";
  }

  list target {
    key name;
    leaf name {
      type snmp:identifier;
      description
        "The name identifies the target.";
    }
    leaf transport {
      type snmp-transport-type;
```

```
      description
        "The transport protocol used.";
    }
    leaf target-parmas {
      type snmp:identifier;
      description
        "Parameters for the target.";
    }
    description
      "The list of targets.";
    reference
      "RFC 3413 & RFC 7407";
  }

  list target-params {
    key name;
    leaf name {
      type snmp:identifier;
      description
        "The name identifies the target params.";
    }
    choice params {
      case usm {
        uses snmp:usm-target-params;
        description
          "Reuse the grouping defined in ietf-snmp-usm";
      }
      case tsm {
        if-feature snmp:tsm;
        uses snmp:tsm-target-params;
        description
          "Reuse the grouping defined in ietf-snmp-tsm";
      }
      description
        "The parameters specific to each security model.";
    }
    description
      "List of target parameters.";
  }

  container vacm {
    leaf vacm-enable {
      type boolean;
      config false;
      description
        "Whether VACM based security configurations are used.";
    }
    list group {
```

```
      key name;
      leaf name {
        type snmp:group-name;
        description
          "The name of this VACM group.";
      }
      list member {
        key "security-name";
        leaf security-name {
          type snmp:security-name;
          description
            "The securityName of a group member.";
        }
        leaf-list security-model {
          type snmp:security-model;
          min-elements 1;
          description
            "The security models under which this security-name is a member of t
his group.";
        }
        description
          "A member of this VACM group.";
      }
      list access {
        key "context security-model security-level";
        leaf context {
          type snmp:context-name;
          description
            "The context under which the access rights apply.";
        }
        leaf context-match {
          type enumeration {
            enum exact {
              value 1;
              description
                "The context match type: exact.";
            }
            enum prefix {
              value 2;
              description
                "The context match type: prefix";
            }
          }
          description
            "The match type of the context.";
        }
        leaf security-model {
          type snmp:security-model-or-any;
          description
```

```
            "The security model under which the access rights apply.";
          }
          leaf security-level {
            type snmp:security-level;
            description
              "The minimum security level under which the access rights apply.";
          }
          leaf read-view {
            type snmp:view-name;
            description
              "The name of the MIB view of the SNMP context authorizing read acces
s. If this leaf does not exist in a configuration, it maps to a zero-length vacm
AccessReadViewName.";
          }
          leaf wirte-view {
            type snmp:view-name;
            description
              "The name of the MIB view of the SNMP context authorizing write acce
ss. If this leaf does not exist in a configuration, it maps to a zero-length vac
mAccessWriteViewName.";
          }
          leaf notify-view {
            type snmp:view-name;
            description
              "The name of the MIB view of the SNMP context authorizing notify acc
ess. If this leaf does not exist in a configuration, it maps to a zero-length va
cmAccessNotifyViewName.";
          }
          description
            "Definition of access right for groups.";
        }
        description
          "VACM groups";
        reference
          "RFC 3415: View-based Access Control Model (VACM) for the Simple Network
 Management Protocol (SNMP)";
      }
      list view {
        key name;
        leaf name {
          type snmp:view-name;
          description
            "The name of this MIB view.";
        }
        leaf-list include {
          type snmp:wildcard-object-identifier;
          description
            "A family of subtrees included in this MIB view.";
        }
        leaf-list exclude {
          type snmp:wildcard-object-identifier;
          description
            "A family of subtrees excluded in this MIB view.";
        }
        description
```

```
        "Definition of MIB views.";
    }
    description
      "The security configurations for View-based Access Control Model (VACM).";
  }

  container usm {
    leaf usm-enable {
      type boolean;
      config false;
      description
        "Whether USM based security configurations are used.";
    }
    container local {
      uses user-auth-priv;
      description
        "A list of local users and their corresponding authentication and privac
y protocols.";
    }
    container remote {
      uses user-auth-priv;
      description
        "A list of remote users and their corresponding authentication and priva
cy protocols.";
    }
    description
      "Configuration of the User-based Security Model.";
  }

  container tsm {
    if-feature tsm;
    leaf tsm-enable {
      type boolean;
      config false;
      description
        "Whether TSM based security configurations are used.";
    }
    description
      "Configuration of Transport Security Model.";
  }
}
<CODE ENDS>
```

6.6. Module 'ietf-netconf-security'

```
module ietf-netconf-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-security";
  prefix netsec;
```

```
  import ietf-admin-access-security {
    prefix accsec;
  }

  import ietf-inet-types {
    prefix inet;
        reference
      "RFC 6991: Common YANG Data Types";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-netconf-security YANG module.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  /*
  * features
  */
  feature listen {
    description
      "The 'listen' feature indicates that the NETCONF server supports opening a
 port to accept NETCONF client connections using at least one transport (e.g., S
SH, TLS, etc.).";
  }

  feature ssh-listen {
    description
      "The 'ssh-listen' feature indicates that the NETCONF server supports openi
ng a port to accept NETCONF over SSH client connections.";
    reference
      "RFC 6242: Using the NETCONF Protocol over Secure Shell (SSH)";
  }
```

```
  feature tls-listen {
    description
      "The 'tls-listen' feature indicates that the NETCONF server supports openi
ng a port to accept NETCONF over TLS client connections.";
    reference
      "RFC 7589: Using the NETCONF Protocol over Transport Layer Security (TLS)
with Mutual X.509 Authentication";
  }

  feature call-home {
    description
      "The 'call-home' feature indicates that the NETCONF server supports initia
ting NETCONF call home connections to NETCONF clients using at least one transpo
rt (e.g., SSH, TLS, etc.).";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  feature ssh-call-home {
    description
      "The 'ssh-call-home' feature indicates that the NETCONF server supports in
itiating a NETCONF over SSH call home connection to NETCONF clients.";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  feature tls-call-home {
    description
      "The 'tls-call-home' feature indicates that the NETCONF server supports in
itiating a NETCONF over TLS call home connection to NETCONF clients.";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  /*
  * leaf & containers
  */

  leaf netconf-enable {
    type boolean;
    description
      "Whether the NETCONF protocol is used.";
  }

  container listen {
    if-feature listen;
    list endpoint {
      key name;
      leaf name {
        type string;
        description
          "The name of the NETCONF listen endpoint.";
      }
      choice transport {
```

```
        case ssh {
          if-feature ssh-listen;
          leaf port {
            type inet:port-number;
            description
              "The local port number to listen on.";
          }
          uses accsec:ssh-server-attribute-grouping;
          description
            "SSH based listening.";
        }
        case tls {
          if-feature tls-listen;
          leaf port {
            type inet:port-number;
            description
              "The local port number to listen on.";
          }
          uses accsec:tls-server-attribute-grouping;
          description
            "TLS based listening.";
        }
        description
          "The transport protocol used.";
      }
      description
        "List of endpoints to listen for NETCONF connections.";
    }
    description
      "Configurations related the listen behavior.";
  }

  container call-home {
    if-feature call-home;
    list netconf-client {
      key name;
      leaf name {
        type string;
        description
          "The name of the remote NETCONF client.";
      }
      container endpoints {
        list endpoint {
          key name;
          leaf name {
            type string;
            description
              "The name for this endpoint.";
```

```
          }
          choice transport {
            case ssh {
              if-feature ssh-call-home;
              leaf port {
                type inet:port-number;
                description
                  "The IP port for this endpoint.";
              }
              uses accsec:ssh-server-attribute-grouping;
              description
                "SSH based call-home.";
            }
            case tls {
              if-feature tls-call-home;
              leaf port {
                type inet:port-number;
                description
                  "The IP port for this endpoint.";
              }
              uses accsec:tls-server-attribute-grouping;
              description
                "TLS based call-home.";
            }
            description
              "The used transport protocol.";
          }
          description
            "A list of endpoints for this NETCONF server to try to connect in se
quence.";
        }
        description
          "List of endpoints";
      }
      description
        "List of NETCONF clients the NETCONF server is to initiate call-home con
nections to in parallel.";
    }
    description
      "Configurations related to call-home behavior.";
  }
}
```

6.7. Module 'ietf-port-management-security'

```
<CODE BEGINS> file "ietf-port-management-security@2018-10-16.yang"
module ietf-port-management-security {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-port-management-security";
  prefix acsec;
```

```
  import ietf-inet-types {
    prefix inet;
        reference
      "RFC 6991: Common YANG Data Types";
  }

  organization
    "IETF SACM (Security Automation and Continuous Monitoring) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/sacm/
    WG List: sacm@ietf.org

    Editor: Qiushi Lin
            linqiushi@huawei.com;
    Editor: Liang Xia
            frank.xialiang@huawei.com
        Editor: Henk Birkholz
            henk.birkholz@sit.fraunhofer.de";

  description
    "This YANG module defines ietf-port-management-security YANG module.";

  revision 2018-10-16 {
    description "Initial version.";
        reference
      "draft-lin-sacm-nid-mp-security-baseline-04: The Data Model of Network Inf
rastructure Device Management Plane Security Baseline";
  }

  list port-list {
    key port-number;
    leaf port-number {
      type inet:port-number;
      description
        "The port number.";
    }
    leaf port-status {
      type boolean;
      description
        "The status of the port: open or shut-down.";
    }
    description
      "The status of all the ports in the device.";
  }
}
<CODE ENDS>
```

7. Acknowledgements

8. IANA Considerations

This document requires no IANA actions.

9. Security Considerations

Secure transport should be used to retrieve the current status of management plane security baseline.

10. References

10.1. Normative References

[I-D.birkholz-sacm-yang-content]
Birkholz, H. and N. Cam-Winget, "YANG subscribed notifications via SACM Statements", draft-birkholz-sacm-yang-content-01 (work in progress), January 2018.

[I-D.dong-sacm-nid-cp-security-baseline]
Dong, Y. and L. Xia, "The Data Model of Network Infrastructure Device Control Plane Security Baseline", draft-dong-sacm-nid-cp-security-baseline-00 (work in progress), September 2017.

[I-D.dong-sacm-nid-infra-security-baseline]
Dong, Y. and L. Xia, "The Data Model of Network Infrastructure Device Infrastructure Layer Security Baseline", draft-dong-sacm-nid-infra-security-baseline-01 (work in progress), May 2018.

[I-D.ietf-netconf-keystore]
Watsen, K., "YANG Data Model for a Centralized Keystore Mechanism", draft-ietf-netconf-keystore-06 (work in progress), September 2018.

[I-D.ietf-netconf-netconf-client-server]
Watsen, K., "NETCONF Client and Server Models", draft-ietf-netconf-netconf-client-server-07 (work in progress), September 2018.

[I-D.ietf-netconf-ssh-client-server]
Watsen, K. and G. Wu, "YANG Groupings for SSH Clients and SSH Servers", draft-ietf-netconf-ssh-client-server-07 (work in progress), September 2018.

[I-D.ietf-netconf-tls-client-server]
Watsen, K. and G. Wu, "YANG Groupings for TLS Clients and TLS Servers", draft-ietf-netconf-tls-client-server-07 (work in progress), September 2018.

[I-D.ietf-netmod-acl-model]
Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-20 (work in progress), October 2018.

[I-D.ietf-netmod-syslog-model]
Wildes, C. and K. Koushik, "A YANG Data Model for Syslog Configuration", draft-ietf-netmod-syslog-model-26 (work in progress), March 2018.

[I-D.ietf-sacm-information-model]
Waltermire, D., Watson, K., Kahn, C., Lorenzin, L., Cokus, M., Haynes, D., and H. Birkholz, "SACM Information Model", draft-ietf-sacm-information-model-10 (work in progress), April 2017.

[I-D.xia-sacm-nid-dp-security-baseline]
Xia, L. and G. Zheng, "The Data Model of Network Infrastructure Device Data Plane Security Baseline", draft-xia-sacm-nid-dp-security-baseline-02 (work in progress), June 2018.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC7407] Bjorklund, M. and J. Schoenwaelder, "A YANG Data Model for SNMP Configuration", RFC 7407, DOI 10.17487/RFC7407, December 2014, <https://www.rfc-editor.org/info/rfc7407>.

10.2. Informative References

[I-D.ietf-tls-oldversions-deprecate]
Moriarty, K. and S. Farrell, "Deprecating TLSv1.0 and TLSv1.1", draft-ietf-tls-oldversions-deprecate-00 (work in progress), September 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

Appendix A.

The following is the whole structure of the YANG tree diagram for network infrastructure device management plane. The existed RFCs and drafts that related this document are listed at the right side.

Modules	Related RFCs/Drafts
ietf-admin-account-security	None
ietf-admin-access-security	draft-ietf-netconf-keystore,draft-ietf-netconf-ssh-client-server,draft-ietf-netconf-tls-client-server
ietf-aaa-security	RFC7317
ietf-admin-access-statistics	None
ietf-snmp-security	RFC7407
ietf-netconf-security	draft-ietf-netconf-netconf-client-server,draft-ietf-netconf-keystore
ietf-port-management-security	None
ietf-log-security	draft-ietf-netmod-syslog-model
ietf-file-security	draft-ietf-netconf-keystore,draft-ietf-netconf-ssh-client-server,draft-ietf-netconf-tls-client-server

The modules defined in this document and related RFCs/drafts

Draft [I-D.ietf-netconf-tls-client-server] and draft [I-D.ietf-netconf-ssh-client-server] focus on YANG models for TLS-specific configuration and SSH-specific configuration respectively. The transport-level configuration, such as what ports to listen-on or connect-to, is not included. Besides, as these grouping focus on configurations, the configuration of private-key and "certificate-expiration" notification are not needed. Draft [I-D.ietf-netconf-netconf-client-server] defines NETCONF YANG model based on the data models defined in the above two documents.

[RFC7317] defines a YANG data model for system management of device containing a NETCONF sever. It summarizes data modules for NETCONF user authentication, and defined YANG module for client to configure the RADIUS authentication server information. Three methods are defined for user authentication: public key for local users over SSH, password for local users over any secure transport, password for RADIUS users over any secure transport.

[RFC7407] defines a YANG model for SNMP configuration it is not limited security related configurations and status.

Draft [I-D.ietf-netmod-syslog-model] defines a YANG model for Syslog configuration, including TLS based transport security and syslog messages signing.

Authors' Addresses

Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen, Guangdong 518129
China

Email: linqiushi@huawei.com

Liang Xia
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: Frank.xialiang@huawei.com

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
Darmstadt 64295
Germany

Email: henk.birkholz@sit.fraunhofer.de

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2019

L. Xia
G. Zheng
W. Pan
Huawei

October 22, 2018

The Data Model of Network Infrastructure Device Data Plane Security Baseline
draft-xia-sacm-nid-dp-security-baseline-03

Abstract

This document proposes one part of the security baseline YANG for network infrastructure device (i.e., router, switch, firewall, etc.): data plane security baseline. The companion documents [I-D.ietf-lin-sacm-nid-mp-security-baseline], [I- D.ietf-dong-sacm-nid-infra-security-baseline] cover other parts of the security baseline YANG for network infrastructure device respectively: management plane security baseline, infrastructure layer security baseline.

Table of Contents

1. Introduction

1.1. Objective

Network security is an essential part of the overall network deployment and operation. Due to the following reasons, network infrastructure devices (e.g. switch, router, firewall) are always the objective and exploited by the network attackers, which bring damages to the victim network:

- The existence of a lot of unsafe access channels: for the history reason, some old and unsafe protocols still run in the network devices, like: SNMP v1/v2, Telnet, etc., and are not mandatory to be replaced by the according safer protocols (SNMP v3, SSH). Attackers easily exploit them for attack (e.g., invalid login, message eavesdropping);

- The openness nature of TCP/IP network: despite the benefits of network architecture design and connectivity brought by the network openness, a lot of threats exist at the same time. Spoofing address, security weakness for various protocols, traffic flooding, and other kinds of threat are originated from the network openness;

- The security challenge by the network complexity: network are becoming more complex, with massive nodes, various protocols and flexible topology. Without careful design and strict management, as well as operation automation, the policy consistency of network security management cannot be ensured. It's common that part of the network infrastructure is subject to attack;

- The complex functionality of device: the complexity of device itself increases the difficulty of carrying out the security hardening measurements, as well as the skill requirements to the network administrator. As a result, the network administrator may not be capable of or willing to realize all the security measurements, in addition to implementing the other basic functionalities;

- The capacity and capability mismatching between the data plane and the control plane: there are a large mismatching of the traffic processing capacity and capability between different planes. Without effective control, the large volume of traffic from the data plane will flooding attack the other planes easily.

Therefore, the importance of ensuring the security of the network infrastructure devices is out of question. To secure the network infrastructure devices, one important task is to identify as far as possible the threats and vulnerabilities in the device itself, such as: unnecessary services, insecure configurations, abnormal status, etc., then enforce the corresponding security hardening measurements, such as: update the patch, modify the security configuration, enhance the security mechanism, etc.. We call this task the developing and deploying the security baseline for the network infrastructure, which provides a solid foundation for the overall network security. This document aims to describe the security baseline for the network infrastructure, which is called security baseline in short in this document.

1.2. Security Baseline

Basically, security baseline can be designed and deployed into different layers of the devices:

- application layer: refers to the application platform security solution and the typical application security mechanisms it provided like: identity authentication, access control, permission management, encryption and decryption, auditing and tracking, privacy protection, to ensure secure application data transmission/exchange, secure storage, secure processing, ensuring the secure operation of the application system. Specific examples may be: web application security, software integrity protection, encryption of sensitive data, privacy protection, and lawful interception interfaces and secure third-party component;

- network layer: refers to a series of security measures, to protect the network resources and network services running on the device network platform. Network layer security over network product is complicated. Therefore, it is divided into data plane, control plane, management plane to consider:

  * data plane: focus on the security hardening configuration and status to protect the data plane traffic against eavesdropping, tampering, forging and flooding attacking the network;

  * control plane: focus on the control signaling security of the network infrastructure device, to protect their normal exchange against various attacks (i.e., eavesdropping, tampering, forging and flooding attack) and restrict the malicious control signaling, for ensuring the correct network topology and forwarding behavior;

  * management plane: focus on the management information and platform security. More specific, it includes all the security configuration and status involved in the network OAM process;

- infrastructure layer: refers to all the security design about the device itself and its running OS. As the foundation of the upper layer services, the secure infrastructure layer must be assured. The specific mechanisms include: OS security, key management, cryptography security, certificate management, software integrity.

1.3. Security Baseline Data Model Design

The security baseline varies according to many factors, like: different device types (i.e., router, switch, firewall), the supporting security features of device, the specific security requirements of network operator. It’s impossible to design a complete set for it, so this document and the companion ones are going to propose the most important and universal points of them. More baseline contents can be added in future following the data model scheme specified.

[I-D.ietf-birkholz-sacm-yang-content] defines a method of constructing the YANG data model scheme for the security posture assessment of the network infrastructure device by brokering of YANG push telemetry via SACM statements. The basic steps are:

- use YANG push mechanism[I-D.ietf-netconf-yang-push]to collect the created streams of notifications (telemetry) [I-D.ietf-netconf-subscribed-notifications]providing SACM content on SACM data plane, and the filter expressions used in the context of YANG subscriptions constitute SACM content that is imperative guidance consumed by SACM components on SACM management plane;

- then encapsulate the above YANG push output into a SACM Content Element envelope, which is again encapsulated in a SACM statement envelope;

- lastly, publish the SACM statement into a SACM domain via xmpp-grid publisher.

In this document, we follow the same way as [I-D.ietf-birkholz-sacm-yang-content] to define the YANG output for network infrastructure device security baseline posture based on the SACM information model definition [I-D.ietf-sacm-information-model].

1.4. Summary

The following contents propose part of the security baseline YANG output for network infrastructure device: data plane security baseline. The companion documents [I-D.ietf- dong-sacm-nid-cp-security-baseline], [I-D.ietf-lin-sacm-nid-mp-security-baseline], [I-D.ietf-xia-sacm-nid-app-infr-layers-security-baseline] cover other parts of the security baseline YANG output for network infrastructure device respectively: control plane security baseline, management plane security baseline, application layer and infrastructure layer security baseline.

2. Terminology

2.1. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2.2. Definition of Terms

This document uses the terms defined in [I-D.draft-ietf-sacm-terminology].

3. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node and "*" denotes a "list" and "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

4. Data Model Structure

As the network infrastructure device, it makes decision of the forwarding path based on the IP/MAC address and sends the packet in data plane. The NP or ASIC are the main components for the data plane functions.

This section describes the key data plane security baseline of the network infrastructure devices, and defines their specific data models.

4.1. Layer 2 protection

Mac table is the key resource in terms of layer 2 forwarding, also easily attacked by learning massive invalid mac address. The mac limit function is to protect the mac table by limiting the maximum number of learned mac address in appointed interfaces. The mac address is not learned and the packet is discarded when the up-limit is reached, and the alarm is created possibly.

If the broadcast traffic is not suppressed in layer 2 network (i.e., Ethernet), a great amount of network bandwidth is consumed by a great deal of broadcast traffic. The network performance is degraded, even

interrupting the communication. In such a case, configuring the broadcast traffic suppression on the device to ensure some bandwidth can be reserved for unicast traffic forwarding when broadcast traffic bursts across the network. It's flexible to configure the device to suppress broadcast, multicast, and unknown unicast traffic on an interface, a specified interface in a VLAN, a sub-interface, and over a virtual switch instance (VSI) pseudo wire (PW).

```
module: ietf-layer2-protection
    +--rw mac-limit
    |  +--rw vlan-mac-limits
    |  |  +--rw vlan-mac-limit* [vlan-id]
    |  |     +--rw vlan-id       mac-vlan-id
    |  |     +--rw maximum       uint32
    |  |     +--rw rate?         uint32
    |  |     +--rw action?       mac-limit-forward
    |  |     +--rw alarm?        mac-enable-status
    |  +--rw bd-mac-limits
    |  |  +--rw bd-mac-limit* [bd-id]
    |  |     +--rw bd-id         uint32
    |  |     +--rw maximum       uint32
    |  |     +--rw rate?         uint32
    |  |     +--rw action?       mac-limit-forward
    |  |     +--rw alarm?        mac-enable-status
    |  +--rw vsi-mac-limits
    |  |  +--rw vsi-mac-limit* [vsi-name]
    |  |     +--rw vsi-name      string
    |  |     +--rw maximum       uint32
    |  |     +--rw rate?         uint32
    |  |     +--rw action?       mac-limit-forward
    |  |     +--rw alarm?        mac-enable-status
    |  +--rw pw-mac-limits
    |  |  +--rw pw-mac-limit* [vsi-name pw-name]
    |  |     +--rw vsi-name      string
    |  |     +--rw pw-name       string
    |  |     +--rw maximum       uint32
    |  |     +--rw rate?         uint32
    |  |     +--rw action?       mac-limit-forward
    |  |     +--rw alarm?        mac-enable-status
    |  +--rw if-mac-limits
    |  |  +--rw if-mac-limit* [if-name]
    |  |     +--rw if-name       string
    |  |     +--rw maximum       uint32
    |  |     +--rw rate?         uint32
    |  |     +--rw action?       mac-limit-forward
    |  |     +--rw alarm?        mac-enable-status
    |  +--rw subif-mac-limits
    |     +--rw subif-mac-limit* [if-name]
```

```
|        +--rw if-name       string
|        +--rw maximum       uint32
|        +--rw rate?         uint32
|        +--rw action?       mac-limit-forward
|        +--rw alarm?        mac-enable-status
+--rw traffic-suppress
   +--rw vlan-suppresses
   |  +--rw vlan-suppress* [vlan-id suppress-type direction]
   |     +--rw vlan-id          mac-vlan-id
   |     +--rw suppress-type    suppress-type
   |     +--rw direction        direction-type
   |     +--rw cir?             uint64
   |     +--rw cbs?             uint64
   +--rw bd-suppresses
   |  +--rw bd-suppress* [bd-id suppress-type direction]
   |     +--rw bd-id            uint32
   |     +--rw suppress-type    suppress-type
   |     +--rw direction        direction-type
   |     +--rw cir?             uint64
   |     +--rw cbs?             uint64
   +--rw vsi-suppresses
   |  +--rw vsi-suppress* [vsi-name suppress-type direction]
   |     +--rw vsi-name         string
   |     +--rw suppress-type    suppress-type
   |     +--rw direction        direction-type
   |     +--rw cir?             uint64
   |     +--rw cbs?             uint64
   +--rw pw-suppresses
   |  +--rw pw-suppress* [vsi-name pw-name suppress-type direction]
   |     +--rw vsi-name         string
   |     +--rw pw-name          string
   |     +--rw suppress-type    suppress-type
   |     +--rw direction        direction-type
   |     +--rw cir?             uint64
   |     +--rw cbs?             uint64
   +--rw if-suppresses
   |  +--rw if-suppress* [if-name suppress-type direction]
   |     +--rw if-name          string
   |     +--rw suppress-type    suppress-type
   |     +--rw direction        direction-type
   |     +--rw percent?         uint64
   |     +--rw packets?         uint64
   |     +--rw cir?             uint64
   |     +--rw cbs?             uint64
   +--rw sub-if-suppresses
   |  +--rw sub-if-suppress* [if-name suppress-type direction]
   |     +--rw if-name          string
   |     +--rw suppress-type    suppress-type
```

```
      |     +--rw direction        direction-type
      |     +--rw cir?             uint64
      |     +--rw cbs?             uint64
      +--rw if-storm-controls
         +--rw if-storm-control* [if-name]
            +--rw if-name                          string
            +--rw action?                          storm-ctrl-action-type
            +--rw trap-enable?                     enable-type
            +--rw log-enable?                      enable-type
            +--rw interval?                        uint64
            +--rw if-packet-control-rules
            |  +--rw if-packet-control-rule* [packet-type]
            |     +--rw packet-type                storm-ctrl-type
            |     +--rw rate-type?                 storm-ctrl-rate-type
            |     +--rw min-rate                   uint64
            |     +--rw max-rate                   uint64
            +--ro if-storm-control-infos
               +--ro ifstorm-control-info* [packet-type]
                  +--ro packet-type                storm-ctrl-type
                  +--ro punish-status?             storm-ctrl-action-type
                  +--ro last-punish-time?          string
```

4.2. ARP

ARP security is a set of functions to protect the ARP protocol and networks against malicious attacks so that the network communication keeps stable and important user information is protected, which mainly includes:

ARP anti-spoofing functions: protect devices against spoofing ARP attack packets, improving the security and reliability of network communication.

ARP anti-flooding functions: relieve CPU load and prevent the ARP table overflow, ensuring normal network operation.

```
module: ietf-arp-sec
    +--rw arp-sec
       +--rw arp-packet-validate
       |  +--rw global-validate-type                  arp-validate-type
       |  +--rw if-validate-rule* [if-name]
       |     +--rw if-name                            string
       |     +--rw validate-type                      arp-validate-type
       +--rw arp-gratuitous-control
       |  +--rw global-send-gratuitous-enable         boolean
       |  +--rw global-receive-gratuitous-enable      boolean
       |  +--rw if-gratuitous-rule* [if-name]
       |     +--rw if-name              string
```

```
|     +--rw send-gratuitous-enable              boolean
|     +--rw receive-gratuitous-enable           boolean
+--rw arp-learning-control
|  +--rw global-strict-learning-enable          boolean
|  +--rw if-learning-rule* [if-name]
|     +--rw if-name                             string
|     +--rw learning-disable                    boolean
|     +--rw strict-learning-enable              boolean
+--rw arp-entry-limit
|  +--rw if-limit-rule* [if-name]
|  |  +--rw if-name                             string
|  |  +--rw entry-maximum                       uint32
|  +--rw if-vlan-limit-rule* [if-name vlan-begin]
|     +--rw if-name                             string
|     +--rw vlan-begin                          mac-vlan-id
|     +--rw vlan-end                            mac-vlan-id
|     +--rw entry-maximum                       uint32
+--rw arp-rate-limit
|  +--rw global-rate-limit                      uint32
|  +--rw limit-by-source-mac
|  |  +--rw rate-limit-per-source-mac           uint32
|  |  +--rw source-mac-rule* [source-mac]
|  |     +--rw source-mac                       mac-address
|  |     +--rw rate-limit                       uint32
|  +--rw limit-by-source-ip
|  |  +--rw rate-limit-per-source-ip            uint32
|  |  +--rw source-ip-rule* [source-ip]
|  |     +--rw source-ip                        inet:ip-address
|  |     +--rw rate-limit                       uint32
|  +--rw limit-by-destination-ip
|  |  +--rw rate-limit-per-destination-ip       uint32
|  +--rw limit-by-interface
|  |  +--rw rate-limit-per-interface            uint32
|  |  +--rw interface-rule* [if-name]
|  |     +--rw if-name                          string
|  |     +--rw rate-limit                       uint32
|  +--rw limit-by-vlan
|     +--rw vlan-rule* [vlan-id]
|        +--rw vlan-id                          mac-vlan-id
|        +--rw rate-limit                       uint32
+--rw arp-miss-rate-limit
|  +--rw global-rate-limit                      uint32
|  +--rw limit-by-source-ip
|  |  +--rw rate-limit-per-source-ip            uint32
|  |  +--rw source-ip-rule* [source-ip]
|  |     +--rw source-ip                        inet:ip-address
|  |     +--rw rate-limit                       uint32
|  +--rw limit-by-interface
```

```
|  |  +--rw rate-limit-per-interface            uint32
|  |  +--rw interface-rule* [if-name]
|  |     +--rw if-name                          string
|  |     +--rw rate-limit                       uint32
|  +--rw limit-by-vlan
|     +--rw vlan-rule* [vlan-id]
|        +--rw vlan-id                          mac-vlan-id
|        +--rw rate-limit                       uint32
+--ro sec-dis-arp-chks
   +--ro sec-dis-arp-chk* [slot-id check-type]
      +--ro slot-id                            string
      +--ro check-type                         arp-attack-type
      +--ro total-packets?                     uint64
      +--ro passed-packets?                    uint64
      +--ro dropped-packets?                   uint64
```

4.3. URPF

Unicast Reverse Path Forwarding (URPF) is a technology used to defend against network attacks based on source address spoofing. Generally, upon receiving a packet, a router first obtains the destination IP address of the packet and then searches the forwarding table for a route to the destination address. If the router finds such a route, it forwards the packet; otherwise, it discards the packet. A URPF-enabled router, however, obtains the source IP address of a received packet and searches for a route to the source address. If the router fails to find the route, it considers that the source address is a forged one and discards the packet. In this manner, URPF can effectively protect against malicious attacks that are launched by changing the source addresses of packets.

URPF can be performed in strict or loose mode. The strict mode checks both the existence of source address in the route table and the interface consistency, while loose mode only checks if the source address is in the route table. In some case, the router may have only one default route to the router of the ISP. Therefore, matching the default route entry needs to be supported.

URPF can be performed over interface, defined flow and traffic sent to local CPU.

```
module: ietf-urpf-sec
    +--rw urpf-sec
       +--rw interface-urpf
       |  +--rw interface-rule* [if-name]
       |     +--rw if-name                                  string
       |     +--rw urpf-mode                                urpf-mode-type
       |     +--rw allow-default                            boolean
       +--rw flow-urpf
       augment /policy:policies/policy:policy-entry +
               /policy:classifier-entry +
               /policy:classifier-action-entry-cfg +
               /policy:action-cfg-params:
       |  +--:(urpf)
       |     +--rw urpf-cfg
       |        +--rw urpf-mode                           urpf-mode-type
       |        +--rw allow-default                       boolean
       +--rw local-urpf
          +--rw slot-rule* [slot-id]
             +--rw slot-id                                 string
             +--rw urpf-mode                               urpf-mode-type
             +--rw allow-default                           boolean
```

4.4. DHCP Snooping

DHCP, which is widely used on networks, dynamically assigns IP addresses to clients and manages configuration information in a centralized manner. During DHCP packet forwarding, some attacks may occur, such as bogus DHCP server attacks, DHCP exhaustion attacks, denial of service (DoS) attacks, and DHCP flooding attacks.

DHCP snooping is a DHCP security feature that functions in a similar way to a firewall between DHCP clients and servers. A DHCP-snooping-capable device intercepts DHCP packets and uses information carried in the packets to create a DHCP snooping binding table. This table records hosts' MAC addresses, IP addresses, IP address lease time, VLAN, and interface information. The device uses this table to check the validity of received DHCP packets. If a DHCP packet does not match any entry in this table, the device discards the packet.

Besides the binding table, DHCP snooping has other security features such as trusted interface, max DHCP user limit and whitelist to defend against the bogus DHCP server, DHCP flooding and other fine-grained DHCP attacks.

```
module: ietf-dhcp-snooping
    +--rw dhcp-snooping
       +--rw dhcp-snooping-enable
       |  +--rw global-enable                          boolean
```

```
|  +--rw enable-vlan* [vlan-id]
|  |  +--rw vlan-id                              uint16
|  |  +--rw dhcp-snp-enable                      boolean
|  +--rw enable-vlan-interface* [vlan-id if-name]
|  |  +--rw vlan-id                              uint16
|  |  +--rw if-name                              string
|  |  +--rw dhcp-snp-enable                      boolean
|  +--rw enable-interface* [if-name]
|  |  +--rw if-name                              string
|  |  +--rw dhcp-snp-enable                      boolean
|  +--rw enable-bd* [bd-id]
|     +--rw bd-id                                uint32
|     +--rw dhcp-snp-enable                      boolean
+--rw dhcp-snooping-trust
|  +--rw trust-vlan* [vlan-id]
|  |  +--rw vlan-id                              uint16
|  |  +--rw dhcp-snp-trust                       boolean
|  |  +--rw untrust-reply-alarm-enable           boolean
|  |  +--rw untrust-reply-alarm-threshold        uint32
|  +--rw trust-vlan-interface* [vlan-id if-name]
|  |  +--rw vlan-id                              uint16
|  |  +--rw if-name                              string
|  |  +--rw dhcp-snp-trust                       boolean
|  |  +--rw untrust-reply-alarm-enable           boolean
|  |  +--rw untrust-reply-alarm-threshold        uint32
|  +--rw trust-interface* [if-name]
|  |  +--rw if-name                              string
|  |  +--rw dhcp-snp-trust                       boolean
|  |  +--rw untrust-reply-alarm-enable           boolean
|  |  +--rw untrust-reply-alarm-threshold        uint32
|  +--rw trust-bd* [bd-id]
|     +--rw bd-id                                uint32
|     +--rw dhcp-snp-trust                       boolean
|     +--rw untrust-reply-alarm-enable           boolean
|     +--rw untrust-reply-alarm-threshold        uint32
+--rw dhcp-snooping-packet-check
|  +--rw check-vlan* [vlan-id]
|  |  +--rw vlan-id                              uint16
|  |  +--rw check-vlan-rule* [check-type]
|  |     +--rw check-type                        check-type
|  |     +--rw check-enable                      boolean
|  |     +--rw alarm-enable                      boolean
|  |     +--rw alarm-threshold                   uint32
|  +--rw check-vlan-interface* [vlan-id if-name]
|  |  +--rw vlan-id                              uint16
|  |  +--rw if-name                              string
|  |  +--rw check-vlan-if-rule* [check-type]
|  |     +--rw check-type                        check-type
```

```
|  |  |     +--rw check-enable                        boolean
|  |  |     +--rw alarm-enable                        boolean
|  |  |     +--rw alarm-threshold                     uint32
|  |  +--rw check-interface* [if-name]
|  |  |  +--rw if-name                                string
|  |  |  +--rw check-if-rule* [check-type]
|  |  |     +--rw check-type                          check-type
|  |  |     +--rw check-enable                        boolean
|  |  |     +--rw alarm-enable                        boolean
|  |  |     +--rw alarm-threshold                     uint32
|  |  +--rw check-bd* [bd-id]
|  |     +--rw bd-id                                  uint32
|  |     +--rw check-bd-rule* [check-type]
|  |        +--rw check-type                          check-type
|  |        +--rw check-enable                        boolean
|  |        +--rw alarm-enable                        boolean
|  |        +--rw alarm-threshold                     uint32
|  +--rw dhcp-snooping-max-user-limit
|  |  +--rw user-limit-vlan* [vlan-id]
|  |  |  +--rw vlan-id                                uint16
|  |  |  +--rw max-user-limit                         uint32
|  |  |  +--rw alarm-enable                           boolean
|  |  |  +--rw alarm-threshold                        uint32
|  |  +--rw user-limit-vlan-interface* [vlan-id if-name]
|  |  |  +--rw vlan-id                                uint16
|  |  |  +--rw if-name                                string
|  |  |  +--rw max-user-limit                         uint32
|  |  |  +--rw alarm-enable                           boolean
|  |  |  +--rw alarm-threshold                        uint32
|  |  +--rw user-limit-interface* [if-name]
|  |  |  +--rw if-name                                string
|  |  |  +--rw max-user-limit                         uint32
|  |  |  +--rw alarm-enable                           boolean
|  |  |  +--rw alarm-threshold                        uint32
|  |  +--rw user-limit-bd* [bd-id]
|  |     +--rw bd-id                                  uint32
|  |     +--rw max-user-limit                         uint32
|  |     +--rw alarm-enable                           boolean
|  |     +--rw alarm-threshold                        uint32
|  +--rw dhcp-snooping-rate-limit
|  |  +--rw global-check-enable                       boolean
|  |  +--rw global-rate-limit                         uint32
|  |  +--rw global-alarm-enable                       boolean
|  |  +--rw global-alarm-threshold                    uint32
|  |  +--rw rate-limit-vlan* [vlan-id]
|  |  |  +--rw vlan-id                                uint16
|  |  |  +--rw check-enable                           boolean
|  |  |  +--rw rate-limit                             uint32
```

```
|  |  +--rw alarm-enable                          boolean
|  |  +--rw alarm-threshold                       uint32
|  +--rw rate-limit-vlan-interface* [vlan-id if-name]
|  |  +--rw vlan-id                               uint16
|  |  +--rw if-name                               string
|  |  +--rw check-enable                          boolean
|  |  +--rw rate-limit                            uint32
|  |  +--rw alarm-enable                          boolean
|  |  +--rw alarm-threshold                       uint32
|  +--rw rate-limit-interface* [if-name]
|  |  +--rw if-name                               string
|  |  +--rw check-enable                          boolean
|  |  +--rw rate-limit                            uint32
|  |  +--rw alarm-enable                          boolean
|  |  +--rw alarm-threshold                       uint32
|  +--rw rate-limit-bd* [bd-id]
|     +--rw bd-id                                 uint32
|     +--rw check-enable                          boolean
|     +--rw rate-limit                            uint32
|     +--rw alarm-enable                          boolean
|     +--rw alarm-threshold                       uint32
+--rw dhcp-snooping-static-binding-table
|  +--rw vlan-static-bind-tbl* [vlan-id ip-address ce-vlan]
|  |  +--rw vlan-id                               uint16
|  |  +--rw ip-address                            inet:ip-address
|  |  +--rw mac-address?                          mac-address
|  |  +--rw if-name?                              string
|  |  +--rw ce-vlan                               uint16
|  +--rw if-static-bind-tbl* [if-name ip-address pe-vlan ce-vlan]
|  |  +--rw if-name                               string
|  |  +--rw ip-address                            inet:ip-address
|  |  +--rw mac-address?                          mac-address
|  |  +--rw pe-vlan                               uint16
|  |  +--rw ce-vlan                               uint16
|  +--rw bd-static-bind-tbl* [bd-id ip-address pe-vlan ce-vlan]
|     +--rw bd-id                                 uint32
|     +--rw ip-address                            inet:ip-address
|     +--rw mac-address?                          mac-address
|     +--rw pe-vlan                               uint16
|     +--rw ce-vlan                               uint16
+--rw dhcp-snp-white-lists
|  +--rw dhcp-snp-white-list* [wht-lst-name]
|     +--rw wht-lst-name                          string
|     +--rw apply-flag                            boolean
|     +--rw dhcp-snp-white-rules
|        +--rw dhcp-snp-white-rule* [rule-id]
|           +--rw rule-id                         uint16
|           +--rw src-ip?                         inet:ip-address
```

```
       |           +--rw src-mask?                      inet:ip-address
       |           +--rw dst-ip?                        inet:ip-address
       |           +--rw dst-mask?                      inet:ip-address
       |           +--rw src-port?                      dhcp-snp-port
       |           +--rw dst-port?                      dhcp-snp-port
       +--ro dhcp-snp-dyn-bind-tbls
       |  +--ro dhcp-snp-dyn-bind-tbl* [ip-address outer-vlan inner-vlan vsi-nam
e vpn-name bridge-domain]
       |     +--ro ip-address                          inet:ip-address
       |     +--ro outer-vlan                          uint16
       |     +--ro inner-vlan                          uint16
       |     +--ro vsi-name                            string
       |     +--ro vpn-name                            string
       |     +--ro bridge-domain                       uint32
       |     +--ro mac-address?                        mac-address
       |     +--ro if-name?                            string
       |     +--ro lease?                              yang:date-and-time
       +--ro dhcp-snp-statistics
          +--ro pkt-cnt-drop-by-global-rate             uint32
          +--ro dhcp-snp-vlan-statistics
          |  +--ro dhcp-snp-vlan-statistic* [vlan-id]
          |     +--ro vlan-id                           uint16
          |     +--ro drop-arp-pkt-cnt?                 uint32
          |     +--ro drop-ip-pkt-cnt?                  uint32
          |     +--ro pkt-cnt-drop-by-user-bind?        uint32
          |     +--ro pkt-cnt-drop-by-mac-check?        uint32
          |     +--ro pkt-cnt-drop-by-untrust-reply?    uint32
          |     +--ro pkt-cnt-drop-by-rate?             uint32
          +--ro dhcp-snp-vlan-if-statistics
          |  +--ro dhcp-snp-vlan-if-statistic* [vlan-id if-name]
          |     +--ro vlan-id                           uint16
          |     +--ro if-name                           string
          |     +--ro drop-arp-pkt-cnt?                 uint32
          |     +--ro drop-ip-pkt-cnt?                  uint32
          |     +--ro pkt-cnt-drop-by-user-bind?        uint32
          |     +--ro pkt-cnt-drop-by-mac-check?        uint32
          |     +--ro pkt-cnt-drop-by-untrust-reply?    uint32
          |     +--ro pkt-cnt-drop-by-rate?             uint32
          +--ro dhcp-snp-if-statistics
          |  +--ro dhcp-snp-if-statistic* [if-name]
          |     +--ro if-name                           string
          |     +--ro drop-arp-pkt-cnt?                 uint32
          |     +--ro drop-ip-pkt-cnt?                  uint32
          |     +--ro pkt-cnt-drop-by-user-bind?        uint32
          |     +--ro pkt-cnt-drop-by-mac-check?        uint32
          |     +--ro pkt-cnt-drop-by-untrust-reply?    uint32
          |     +--ro pkt-cnt-drop-by-rate?             uint32
          +--ro dhcp-snp-bd-statistics
             +--ro dhcp-snp-bd-statistic* [if-name]
```

```
               +--ro bd-id                                uint32
               +--ro drop-arp-pkt-cnt?                    uint32
               +--ro drop-ip-pkt-cnt?                     uint32
               +--ro pkt-cnt-drop-by-user-bind?           uint32
               +--ro pkt-cnt-drop-by-mac-check?           uint32
               +--ro pkt-cnt-drop-by-untrust-reply?       uint32
               +--ro pkt-cnt-drop-by-rate?                uint32
```

4.5. CPU Protection

For the network device, there are maybe a large number of packets to be sent to its CPU, or malicious packets attempt to attack the device CPU. If the CPU receives excessive packets, it will be overloaded and support the normal services with very poor performance; In extreme cases, the system fails.

More specifically, services are negatively affected when the CPU is attacked because of the following reasons:

- Valid protocol packets are not distinguished from invalid protocol packets. The CPU is busy in processing a large number of invalid protocol packets. Consequently, the CPU usage rises sharply and valid packets cannot be processed properly

- Packets of some protocols are sent to the CPU through the same channel. When excessive packets of a certain type of protocol packet block the channel, the transmission of other protocol packets is affected

- The bandwidth of a channel is not set appropriately. When an attack occurs, processing of protocol packets on other channels is affected

Accordingly, the following countermeasures can be taken by the network device for CPU protection:

- Collect and classify protocols related to various services running on equipment

- Use ACLs to filter the packets. Valid protocol packets are put into the whitelist and a user-defined flow, other packets are put into the blacklist

- Plan the priorities, channel bandwidth, length of packets, and alarm function of the preceding three lists

- Disable services that are not deployed on the equipment, and control the total forwarding bandwidth

In this manner, the number of packets sent to the CPU is under control, and the bandwidth is ensured preferentially for services with higher priorities. In addition, CPU overload is prevented and an alarm is generated when an attack occurs.

```
module: ietf-cpu-defend
    +--rw cpu-defend
       +--rw cpu-defend-policies
       |  +--rw cpu-defend-policy* [policy-id]
       |     +--rw policy-id                        uint32
       |     +--rw description?                     string
       |     +--rw white-list-acl-number?           uint32
       |     +--rw black-list-acl-number?           uint32
       |     +--rw user-defined-flows
       |     |  +--rw user-defined-flow* [flow-id]
       |     |     +--rw flow-id                    uint32
       |     |     +--rw acl-number                 uint32
       |     +--rw cpu-defend-car-rules
       |        +--rw cpu-defend-car-rule* [rule-type pkt-index flow-id protocol
-type tcp-ip-type]
       |           +--rw rule-type                  rule-type
       |           +--rw pkt-index?                 uint16
       |           +--rw flow-id?                   uint32
       |           +--rw protocol?                  protocol-type
       |           +--rw tcp-ip-type?               tcp-ip-type
       |           +--rw car-attr
       |           |  +--rw cir?                    uint32
       |           |  +--rw cbs?                    uint32
       |           |  +--rw pir?                    uint32
       |           |  +--rw pbs?                    uint32
       |           |  +--rw min-pkt-len?            uint32
       |           |  +--rw pkt-rate?               uint32
       |           |  +--rw weight?                 uint16
       |           +--rw priority?                  priority-type
       |           +--rw drop-alarm
       |              +--rw enable                  boolean
       |              +--rw packets-threshold?      uint32
       |              +--rw interval?               uint16
       |              +--rw speed-threshold?        uint32
       +--rw cpu-defend-policy-bindings
       |  +--rw cpu-defend-policy-binding* [slot-id]
       |     +--rw slot-id                          string
       |     +--rw policy-id                        uint32
       +--ro cpu-defend-cars-cfgs
       |  +--ro cpu-defend-cars-cfg* [slot-id pkt-index]
       |     +--ro slot-id                          string
       |     +--ro pkt-index                        uint16
       |     +--ro cir?                             uint32
       |     +--ro cbs?                             uint32
```

```
|     +--ro min-pkt?                        uint32
|     +--ro priority?                       priority-type
|     +--ro protocol                        protocol-type
+--ro protocol-stats
|  +--ro protocol-stat* [slot-id protocol]
|     +--ro slot-id                         string
|     +--ro protocol                        protocol-type
|     +--ro default-act                     action-type
|     +--ro default-cir                     uint32
|     +--ro default-cbs                     uint32
+--ro sec-non-car-stats
|  +--ro sec-non-car-stat* [slot-id policy-type protocol]
|     +--ro slot-id                         string
|     +--ro policy-type                     policy-type
|     +--ro protocol                        protocol-type
|     +--ro total-packets?                  uint64
|     +--ro passed-packets?                 uint64
|     +--ro dropped-packets?                uint64
+--ro sec-car-stats
|  +--ro sec-car-stat* [slot-id policy-type policy-index]
|     +--ro slot-id                         string
|     +--ro policy-type                     policy-type
|     +--ro policy-index                    uint32
|     +--ro app-enable?                     boolean
|     +--ro app-default-act?                action-type
|     +--ro proto-enable?                   boolean
|     +--ro passed-packets?                 uint64
|     +--ro dropped-packets?                uint64
|     +--ro cfg-cir?                        uint32
|     +--ro cfg-cbs?                        uint32
|     +--ro actual-cir?                     uint32
|     +--ro actual-cbs?                     uint32
|     +--ro priority?                       priority-type
|     +--ro min-pkt-len?                    uint32
|     +--ro acl-deny-packets?               uint64
|     +--ro hist-pps?                       uint64
|     +--ro hist-pps-time?                  yang:date-and-time
|     +--ro average-drop-rate?              uint64
|     +--ro drop-begin-time?                yang:date-and-time
|     +--ro drop-end-time?                  yang:date-and-time
|     +--ro total-dropped-packets?          uint64
+--ro total-packet-stats
|  +--ro total-packet-stat* [slot-id]
|     +--ro slot-id                         string
|     +--ro total-packets?                  uint64
|     +--ro passed-packets?                 uint64
|     +--ro dropped-packets?                uint64
+--rw hostcar-policies
```

```
       |  +--rw hostcar-policy* [slot-id host-car-type]
       |     +--rw slot-id                          string
       |     +--rw host-car-type                    host-car-type
       |     +--rw cir?                             uint32
       |     +--rw pir?                             uint32
       |     +--rw cbs?                             uint32
       |     +--rw pbs?                             uint32
       |     +--rw automatic-adjustment
       |        +--rw enable?                       boolean
       |        +--rw drop-threshold?               uint32
       |        +--rw interval?                     uint32
       +--ro host-car-stats
       |  +--ro host-car-stat* [slot-id host-car-type stat-type host-car-id http
-host-car-id vlan-host-car-id]
       |     +--ro slot-id                          string
       |     +--ro host-car-type                    host-car-type
       |     +--ro stat-type                        stat-type
       |     +--ro host-car-id                      uint32
       |     +--ro http-host-car-id                 uint32
       |     +--ro vlan-host-car-id                 uint32
       |     +--ro passed-bytes?                    uint64
       |     +--ro dropped-bytes?                   uint64
       +--ro host-car-cfgs
          +--ro host-car-cfg* [slot-id]
             +--ro slot-id                         string
             +--ro host-car-type?                  host-car-type-enum
             +--ro default-cir?                    uint32
             +--ro default-pir?                    uint32
             +--ro default-cbs?                    uint32
             +--ro default-pbs?                    uint32
             +--ro actual-cir?                     uint32
             +--ro actual-pir?                     uint32
             +--ro actual-cbs?                     uint32
             +--ro actual-pbs?                     uint32
             +--ro droprate-en?                    boolean
             +--ro log-interval?                   uint32
             +--ro log-threshold?                  uint32
```

4.6. TCP/IP Attack Defense

Defense against TCP/IP attacks is applied to the router on the edge of the network or other routers that are easily to be attacked by illegal TCP/IP packets. Defense against TCP/IP attacks can protect the CPU of the router against malformed packets, fragmented packets, TCP SYN packets, and UDP packets, ensuring that normal services can be processed.

```
module: ietf-tcp-ip-attack-defense
    +--rw tcp-ip-attack-defense
       +--rw anti-enable?                                boolean
       +--rw abnormal-enable?                            boolean
       +--rw udp-flood-enable?                           boolean
       +--rw tcp-syn-enable?                             boolean
       +--rw icmp-flood-enable?                          boolean
       +--rw fragment-enable?                            boolean
       +--rw sec-anti-attack-car-cfg
       |  +--rw cir-fragment?                            uint32
       |  +--rw cir-icmp?                                uint32
       |  +--rw cir-tcp?                                 uint32
       +--rw sec-anti-attack-stats
          +--ro sec-anti-attack-stat* [attack-type]
             +--ro attack-type                           anti-attack-type
             +--ro total-count?                          uint64
             +--ro drop-count?                           uint64
             +--ro pass-count?                           uint64
```

5. Network Infrastructure Device Security Baseline Yang Module

<CODE BEGINS> file "ietf-mac-limit@2018-06-04.yang"

```
module ietf-mac-limit {
namespace "urn:ietf:params:xml:ns:yang:ietf-mac-limit";
prefix mac-limit;
organization
  "IETF SACM Working Group";
contact
  "Liang Xia: Frank.xialiang@huawei.com;
  Guangying Zheng: Zhengguangying@huawei.com";
description
  "MAC address limit.";

revision 2018-06-04 {
  description
    "Init revision";
  reference "xxx.";
}

/*
 * Typedefs
 */
typedef mac-limit-forward {
  type enumeration {
    enum "forward" {
```

```
        description
          "Forward.";
      }
      enum "discard" {
        description
          "Discard.";
      }
    }
    description
      "MAC Limit Forward";
  }
  typedef mac-enable-status {
    type enumeration {
      enum "enable" {
        description
          "Enable.";
      }
      enum "disable" {
        description
          "Disable.";
      }
    }
    description
      "MAC Enable Status";
  }
  typedef mac-vlan-id {
    type uint16 {
      range "1..4094";
    }
    description
      "MAC Vlan Id";
  }
  typedef mac-type {
    type enumeration {
      enum "static" {
        description
          "Static MAC address entry.";
      }
      enum "dynamic" {
        description
          "Dynamic MAC address entry.";
      }
      enum "black-hole" {
        description
          "Blackhole MAC address entry";
      }
      enum "sticky" {
        description
```

```
          "sticky MAC address entry";
        }
        enum "security" {
          description
            "security MAC address entry";
        }
        enum "evn" {
          description
            "EVN MAC address entry.";
        }
        enum "mux" {
          description
            "MUX MAC address entry.";
        }
        enum "snooping" {
          description
            "SNOOPING MAC address entry.";
        }
        enum "tunnel" {
          description
            "TUNNEL MAC address entry.";
        }
        enum "authen" {
          description
            "AUTHEN MAC address entry.";
        }
      }
      description
        "MAC Type";
    }
    typedef suppress-type {
      type enumeration {
        enum "broadcast" {
          description
            "Broadcast.";
        }
        enum "multicast" {
          description
            "Multicast.";
        }
        enum "unknown-unicast" {
          description
            "Unknown unicast.";
        }
        enum "unicast" {
          description
            "Unicast.";
        }
```

```
    }
    description
      "Suppress Type";
  }
  typedef limit-type {
    type enumeration {
      enum "-mac-limit" {
        description
          "Interface MAC rule limit.";
      }
      enum "mac-apply" {
        description
          "Interface MAC rule application.";
      }
    }
    description
      "Limit Type";
  }

  typedef mac-pw-encap-type {
    type enumeration {
      enum "ethernet" {
        description
          "Ethernet.";
      }
      enum "vlan" {
        description
          "VLAN.";
      }
    }
    description
      "MAC PW Encapsulation Type";
  }

  typedef suppress-style {
    type enumeration {
      enum "percent" {
        description
          "Percent.";
      }
      enum "absolute-value" {
        description
          "Absolute value.";
      }
    }
    description
      "Suppress Style";
  }
```

```
  typedef direction-type {
    type enumeration {
      enum "inbound" {
        description
          "Inbound.";
      }
      enum "outbound" {
        description
          "Outbound.";
      }
    }
    description
      "Direction Type";
  }

  typedef storm-ctrl-action-type {
    type enumeration {
      enum "normal" {
        description
          "Normal.";
      }
      enum "error-down" {
        description
          "Error down.";
      }
      enum "block" {
        description
          "Block.";
      }
      enum "suppress" {
        description
          "Suppress";
      }
    }
    description
      "Storm Ctrl Action Type";
  }

  typedef enable-type {
    type enumeration {
      enum "disable" {
        description
          "Disable.";
      }
      enum "enable" {
        description
          "Enable.";
      }
```

```
    }
    description
      "Enable Type";
  }

  typedef storm-ctrl-type {
    type enumeration {
      enum "broadcast" {
        description
          "Broadcast.";
      }
      enum "multicast" {
        description
          "Multicast.";
      }
      enum "unicast" {
        description
          "Unicast.";
      }
      enum "unknown-unicast" {
        description
          "Unknown unicast.";
      }
    }
    description
      "Storm Ctrl Type";
  }

  typedef storm-ctrl-rate-type {
    type enumeration {
      enum "pps" {
        description
          "Packets per second.";
      }
      enum "percent" {
        description
          "Percent.";
      }
      enum "kbps" {
        description
          "Kilo bits per second.";
      }
    }
    description
      "Storm Ctrl Rate Type";
  }
```

```
  container mac {
    description
      "MAC address forwarding. ";
    container mac-limit-rules {
      description
        "Global MAC address learning limit rule.";
      list mac-limit-rule {
        key "rule-name";
        description
          "Global MAC address learning limit.";
        leaf rule-name {
          type string {
            length "1..31";
          }
          description
            "Global MAC address learning limit rule name.";
        }
        leaf maximum {
          type uint32 {
            range "0..131072";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval at which MAC addresses are learned.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number.";
        }
      }
    }
    container vlan-mac-limits {
```

```
      description
        "VLAN MAC address limit list.";
      list vlan-mac-limit {
        key "vlan-id";
        description
          "VLAN MAC address limit.";
        leaf vlan-id {
          type mac-vlan-id;
          description
            "VLAN ID.";
        }
        leaf maximum {
          type uint32 {
            range "0..130048";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned in a VLAN.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval at which MAC addresses are learned in a VLAN.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number in a VLAN.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number in a VLAN.";
        }
      }
    }
    container vsi-mac-limits {
      description
        "VSI MAC address limit list.";
      list vsi-mac-limit {
        key "vsi-name";
        description
          "VSI MAC address limit.";
```

```
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf maximum {
          type uint32 {
            range "0..524288";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned in a VSI.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval at which MAC addresses are learned in a VSI.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number in a VSI.";
        }
        leaf alarm {
          type mac-enable-status;
          default "disable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number in a VSI.";
        }
        leaf up-threshold {
          type uint8 {
            range "80..100";
          }
          mandatory true;
          description
            "Upper limit for the number of MAC addresses.";
        }
        leaf down-threshold {
          type uint8 {
            range "60..100";
          }
          mandatory true;
```

```
          description
            "Upper limit for the number of MAC addresses.";
        }
      }
    }
    container bd-mac-limits {
      description
        "BD MAC address limit list.";
      list bd-mac-limit {
        key "bd-id";
        description
          "BD MAC address limit.";
        leaf bd-id {
          type uint32 {
            range "1..16777215";
          }
          description
            "Specifies the ID of a bridge domain.";
        }
        leaf maximum {
          type uint32 {
            range "0..130048";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned in a BD.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval at which MAC addresses are learned in a BD.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";

          description
            "Forward or discard the packet.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number.";
        }
```

```
      }
    }
    container pw-mac-limits {
      description
        "PW MAC address limit list.";
      list pw-mac-limit {
        key "vsi-name pw-name";
        description
          "PW MAC address limit.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf pw-name {
          type string {
            length "1..15";
          }
          description
            "PW name.";
        }
        leaf maximum {
          type uint32 {
            range "0..130048";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned in a PW.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval at which MAC addresses are learned in a PW.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number in a PW.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
```

```
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number in a PW.";
        }
      }
    }
    container if-mac-limits {
      description
        "Interface MAC address limit list.";
      list if-mac-limit {
        key "if-name limit-type";
        description
          "Interface MAC address limit.";
        leaf if-name {
          type string;
          description
            "Interface name.";
        }
        leaf limit-type {
          type limit-type;
          description
            "Interface MAC limit type.";
        }
        leaf rule-name {
          type leafref {
            path "/mac/mac-limit-rules/mac-limit-rule/rule-name";
          }
          description
            "Rule name.";
        }
        leaf maximum {
          type uint32 {
            range "0..131072";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned on an interface
.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval (ms) at which MAC addresses are learned on an interface.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
```

```
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number on an interface";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number on an interface.";
        }
      }
    }
    container if-vlan-mac-limits {
      description
        "Interface + VLAN MAC address limit list.";
      list if-vlan-mac-limit {
        key "if-name vlan-begin limit-type";
        config false;
        description
          "Interface + VLAN MAC address limit.";
        leaf if-name {
          type string;
          description
            "-name of an interface. ";
        }
        leaf vlan-begin {
          type mac-vlan-id;
          description
            "Start VLAN ID.";
        }
        leaf vlan-end {
          type mac-vlan-id;
          description
            "End VLAN ID.";
        }
        leaf limit-type {
          type limit-type;
          description
            "Interface MAC limit type.";
        }
        leaf rule-name {
          type leafref {
            path "/mac/mac-limit-rules/mac-limit-rule/rule-name";
          }
          description
            "Rule name.";
        }
        leaf maximum {
          type uint32 {
```

```
            range "0..131072";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned on an interface
.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          mandatory true;
          description
            "Interval (ms) at which MAC addresses are learned on an interface.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward the packet.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number.";
        }
      }
    }
    container subif-mac-limits {
      description
        "Sub-interface MAC address limit list.";
      list subif-mac-limit {
        key "if-name limit-type";
        description
          "Sub-interface MAC address limit.";
        leaf if-name {
          type string;
          description
            "-name of a sub-interface. ";
        }
        leaf limit-type {
          type limit-type;
          description
            "Sub-interface MAC limit type.";
        }
        leaf vsi-name {
          type string {
            length "1..36";
```

```
          }
          config false;
          mandatory true;
          description
            "VSI name , EVPN name or bridge domain ID.";
        }
        leaf rule-name {
          type string {
            length "1..31";
          }
          mandatory true;
          description
            "Rule name.";
        }
        leaf maximum {
          type uint32 {
            range "0..131072";
          }
          mandatory true;
          description
            "Maximum number of MAC addresses that can be learned on a sub-interf
ace.";
        }
        leaf rate {
          type uint16 {
            range "0..1000";
          }
          default "0";
          description
            "Interval (ms) at which MAC addresses are learned on a sub-interface
.";
        }
        leaf action {
          type mac-limit-forward;
          default "discard";
          description
            "Discard or forward after the number of learned MAC addresses reache
s the maximum number on a sub-interface.";
        }
        leaf alarm {
          type mac-enable-status;
          default "enable";
          description
            "Whether an alarm is generated after the number of learned MAC addre
sses reaches the maximum number on a sub-interface.";
        }
      }
    }
    container vsi-storm-supps {
      description
        "VSI Suppression List.";
      list vsi-storm-supp {
```

```
        key "vsi-name suppress-type";
        description
          "VSI Suppression.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf suppress-type {
          type suppress-type;
          description
            "Traffic suppression type.";
        }
        leaf cir {
          type uint64 {
            range "0..4294967295";
          }
          default "0";
          description
            "CIR value.";
        }
        leaf cbs {
          type uint64 {
            range "0..4294967295";
          }
          description
            "CBS value.";
        }
      }
    }
    container vlan-storm-supps {
      description
        "VLAN Suppression List.";
      list vlan-storm-supp {
        key "vlan-id suppress-type";
        description
          "VLAN Suppression.";
        leaf vlan-id {
          type mac-vlan-id;
          description
            "VLAN ID.";
        }
        leaf suppress-type {
          type suppress-type;
          description
            "Traffic suppression type.";
```

```
        }
        leaf cir {
          type uint64 {
            range "64..4294967295";
          }
          default "64";
          description
            "CIR value.";
        }
        leaf cbs {
          type uint64 {
            range "10000..4294967295";
          }
          description
            "CBS value.";
        }
      }
    }
    container sub-if-suppresss {
      description
        "Sub-interface traffic suppression list.";
      list sub-if-suppress {
        key "if-name suppress-type direction";
        description
          "Sub-Interface traffic suppression.";
        leaf if-name {
          type string;
          description
            "Sub-interface name.";
        }
        leaf suppress-type {
          type suppress-type;
          description
            "Suppression type.";
        }
        leaf direction {
          type direction-type;
          description
            "Suppression direction.";
        }
        leaf cir {
          type uint64 {
            range "0..4294967295";
          }
          default "0";
          description
            "CIR value.";
        }
```

```
        leaf cbs {
          type uint64 {
            range "0..4294967295";
          }
          description
            "CBS value.";
        }
      }
    }
    container pw-suppresss {
      description
        "PW traffic suppress list.";
      list pw-suppress {
        key "vsi-name pw-name suppress-type";
        description
          "PW traffic suppression.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf pw-name {
          type string {
            length "1..15";
          }
          description
            "PW name.";
        }
        leaf suppress-type {
          type suppress-type;
          description
            "Traffic suppression type.";
        }
        leaf cir {
          type uint64 {
            range "100..4294967295";
          }
          default "100";
          description
            "CIR value.";
        }
        leaf cbs {
          type uint64 {
            range "100..4294967295";
          }
          description
```

```
            "CBS value.";
        }
      }
    }

    container vsi-in-suppressions {
      description
        "VSI inbound traffic suppression list.";
      list vsi-in-suppression {
        key "vsi-name";
        description
          "VSI inbound traffic suppression.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf inbound-supp {
          type mac-enable-status;
          default "enable";
          description
            "Inbound suppression.";
        }
      }
    }
    container vsi-out-suppressions {
      description
        "VSI outbound traffic suppression list.";
      list vsi-out-suppression {
        key "vsi-name";
        description
          "VSI outbound traffic suppression.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          description
            "VSI name.";
        }
        leaf out-bound-supp {
          type mac-enable-status;
          default "enable";
          description
            "Outbound suppression.";
        }
      }
```

```
    }
    container vsi-suppresss {
      description
        "VSI traffic suppression list.";
      list vsi-suppress {
        key "sub-if-name";
        description
          "VSI traffic suppression.";
        leaf vsi-name {
          type string {
            length "1..31";
          }
          mandatory true;
          description
            "VSI name.";
        }

        leaf sub-if-name {
          type string;
          description
            "Sub-interface name.";
        }
        leaf is-enable {
          type boolean;
          default "true";
          description
            "Enable status.";
        }
        leaf suppress-type {
          type suppress-style;
          default "percent";
          description
            "Traffic suppression type.";
        }
        leaf broadcast {
          type uint32 {
            range "0..200000000";
          }
          default "64";
          description
            "Broadcast suppression (kbit/s)";
        }
        leaf broadcast-percent {
          type uint32 {
            range "0..100";
          }
          default "1";
          description
```

```
            "Broadcast suppression.";
        }
        leaf unicast {
          type uint32 {
            range "0..200000000";
          }
          default "64";
          description
            "Unknown unicast suppression (kbit/s).";
        }
        leaf unicast-percent {
          type uint32 {
            range "0..100";
          }
          default "1";
          description
            "Unknown unicast suppression.";

        }
        leaf multicast {
          type uint32 {
            range "0..200000000";
          }
          default "64";
          description
            "Multicast suppression (kbit/s).";
        }
        leaf multicast-percent {
          type uint32 {
            range "0..100";
          }
          default "1";
          description
            "Multicast suppression.";
        }
      }
    }
    container vsi-total-numbers {
      description
        "List of MAC address total numbers in a VSI.";
      list vsi-total-number {
        key "vsi-name slot-id mac-type";
        config false;
        description
          "Total number of MAC addresses in a VSI.";
        leaf vsi-name {
          type string {
            length "1..31";
```

```
          }
          description
            "VSI name.";
        }
        leaf slot-id {
          type string {
            length "1..24";
          }
          description
            "Slot ID.";
        }
        leaf mac-type {
          type mac-type;
          description
            "MAC address type.";
        }
        leaf number {
          type uint32;
          mandatory true;
          description
            "Number of MAC addresses.";
        }
      }
    }
    container if-storm-supps {
      description
        "Interface traffic suppression list.";
      list if-storm-supp {
        key "if-name suppress-type";
        description
          "Interface traffic suppression.";
        leaf if-name {
          type string;
          description
            "-name of an interface. ";
        }
        leaf suppress-type {
          type suppress-type;
          description
            "Suppression type.";
        }
        leaf percent {
          type uint64 {
            range "0..99";
          }
          description
            "Percent.";
        }
```

```
        leaf packets {
          type uint64 {
            range "0..148810000";
          }
          description
            "Packets per second.";
        }
        leaf cir {
          type uint64 {
            range "0..100000000";
          }
          description
            "CIR(Kbit/s).";
        }
        leaf cbs {
          type uint64 {
            range "10000..4294967295";
          }
          description
            "CBS(Bytes).";
        }
      }
    }
    container if-storm-blocks {
      description
        "Interface traffic block list.";
      list if-storm-block {
        key "if-name block-type direction";
        description
          "Interface traffic suppression.";
        leaf if-name {
          type string;
          description
            "-name of an interface. ";
        }
        leaf block-type {
          type suppress-type;
          description
            "Block type.";
        }
        leaf direction {
          type direction-type;
          description
            "Direction.";
        }
      }
    }
    container if-storm-contrls {
```

```
      description
        "Interface storm control list.";
      list if-storm-contrl {
        key "if-name";
        description
          "Interface storm control.";
        leaf if-name {
          type string;
          description
            "-name of an interface. ";
        }
        leaf action {
          type storm-ctrl-action-type;
          default "normal";
          description
            "Action type.";
        }
        leaf trap-enable {

          type enable-type;
          default "disable";
          description
            "Trap state.";
        }
        leaf log-enable {
          type enable-type;
          default "disable";
          description
            "Log state.";
        }
        leaf interval {
          type uint64 {
            range "1..180";
          }
          default "5";
          description
            "Detect interval.";
        }
        container if-packet-contrl-attributes {
          description
            "Storm control rate list.";
          list if-packet-contrl-attribute {
            key "packet-type";
            description
              "Storm control rate.";
            leaf packet-type {
              type storm-ctrl-type;
              description
```

```
            "Packet type.";
          }
          leaf rate-type {
            type storm-ctrl-rate-type;
            default "pps";
            description
              "Storm control rate type.";
          }
          leaf min-rate {
            type uint32 {
              range "1..148810000";
            }
            mandatory true;
            description
              "Storm control min rate.";
          }
          leaf max-rate {
            type uint64 {
              range "1..148810000";
            }
            mandatory true;
            description
              "Storm control max rate.";
          }
        }
      }
      container ifstorm-contrl-infos {
        description
          "Storm control info list.";
        list ifstorm-contrl-info {
          key "packet-type";
          config false;
          description
            "Storm control info";
          leaf packet-type {
            type storm-ctrl-type;
            description
              "Packet type.";
          }
          leaf punish-status {
            type storm-ctrl-action-type;
            description
              "Storm control status.";
          }
          leaf last-punish-time {
            type string {
              length "1..50";
            }
```

```
              description
                "Last punish time.";
            }
          }
        }
      }
    }
  }
}
```

<CODE ENDS>

6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

To be added.

8. Acknowledgements

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[I-D.ietf-netconf-subscribed-notifications]
Voit, E., Clemm, A., Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Customized Subscriptions to a Publisher's Event Streams", draft-ietf-netconf-subscribed-notifications-17 (work in progress), September 2018.

[I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", draft-ietf-netconf-yang-push-19 (work in progress), September 2018.

[I-D.ietf-sacm-information-model]
Waltermire, D., Watson, K., Kahn, C., Lorenzin, L., Cokus, M., Haynes, D., and H. Birkholz, "SACM Information Model", draft-ietf-sacm-information-model-10 (work in progress), April 2017.

Authors' Addresses

Liang Xia
Huawei

Email: frank.xialiang@huawei.com

Guangying Zheng
Huawei

Email: zhengguangying@huawei.com

Wei Pan
Huawei

Email: william.panwei@huawei.com