Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 4, 2011

A. Bierman
Brocade
M. Bjorklund
Tail-f Systems

June 2, 2011

YANG Data Model for System Management
draft-bierman-netmod-system-mgmt-00

Abstract

This document defines a YANG data model for the configuration and identification of the management system of a device.

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 http://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 December 4, 2011.

Copyright Notice

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] data model for the configuration and identification of the management system of a device.

Devices which are managed by NETCONF and perhaps other mechanisms have common properties which need to be configured and monitored in a standard way.

The YANG module defined in this document provides the following features:

- system administrative data configuration
- system identification monitoring
- system time-of-day configuration and monitoring
- user authentication configuration
- local users configuration

1.1. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119].

1.1.1. Terms

The following terms are used within this document:

- system: This term refers to the embodiment of the entire set of management interfaces that a single NETCONF server is supporting at a given moment. The set of physical entities managed by a single NETCONF server can be static or it can change dynamically.

2. Objectives

2.1. System Identification

There are many common properties used to identify devices, operating systems, software versions, etc. that need to be supported in the system data module. These objects are defined as operational data and intended to be specific to the device vendor.

Some user-configurable administrative strings are also provided such as the system location and description.

2.2. System Time Management

The management of the date and time used by the system must be supported. Use of one or more NTP servers to automatically set the system date and time must be possible. Utilization of the Timezone database [I-D.lear-iana-timezone-database] must also be supported.

2.3. User Authentication

The authentication mechanism must support password authentication over RADIUS, to support deployment scenarios with centralized authentication servers. Additionally, local users must be supported, for scenarios when no centralized authentication server exists, or for situations where the centralized authentication server cannot be reached from the device.

Since the mandatory transport protocol for NETCONF is SSH [I-D.ietf-netconf-rfc4742bis] the authentication model must support SSH's "publickey" and "password" authentication methods [RFC4252].

The model for authentication configuration should be flexible enough to support authentication methods defined by other standard documents or by vendors.

3. System Data Model

[FIXME: this section currently just talks about authentication. Add description of the rest of the data model, like we do in snmp-cfg? Otherwise, rename this section to Authentication ...]

[FIXME: introduce a set of submodules to allow for future enhancements of the system data model?]

3.1. User Authentication Model

This document defines three authentication methods for use with NETCONF:

- publickey for local users over SSH
- password for local users over any transport
- password for RADIUS users over any transport

Additional methods may be defined by other standard documents or by vendors.

This document defines two optional YANG features, 'local-users' and 'radius', which the server advertises to indicate support for configuring local users on the device, and for configuring RADIUS access, respectively.

The authentication parameters defined in this document are primarily used to configure authentication of NETCONF users, but MAY also be used by other interfaces, e.g., a Command Line Interface or a Web-based User Interface.

3.2. SSH Public Key Authentication

If the NETCONF server advertises the 'local-users' feature, configuration of local users and their SSH public keys is supported in the /system/authentication/user list.

Public key authentication is requested by the SSH client. The SSH server looks up the user name provided by the client in the /system/authentication/user list, and verifies the key as described in [RFC4253].

If the 'local-users' feature is supported, then when a NETCONF client starts an SSH session towards the server, using the "publickey" authentication 'method name' [RFC4252], the SSH server looks up the user name given in the SSH authentication request in the /system/

authentication/user list,

3.3. Local User Password Authentication

If the NETCONF server advertises the 'local-users' feature, configuration of local users and their passwords is supported in the /system/authentication/user list.

For NETCONF transport protocols that support password authentication, the leaf-list 'user-authentication-order' is used to control if local user password authentication should be used.

In SSH, password authentication is requested by the client. Other NETCONF transport protocols may also support password authentication.

When local user password authentication is requested, the NETCONF transport looks up the user name provided by the client in the /system/ authentication/user list, and verifies the password.

3.4. RADIUS Password Authentication

If the NETCONF server advertises the 'radius' feature, the device supports user authentication RADIUS.

For NETCONF transport protocols that support password authentication, the leaf-list 'user-authentication-order' is used to control if RADIUS password authentication should be used.

In SSH, password authentication is requested by the client. Other NETCONF transport protocols may also support password authentication.

4. System YANG module

RFC Ed.: update the date below with the date of RFC publication and remove this note.

This YANG module imports YANG extensions from ... and references ... [Editor's Note: add proper references]

<CODE BEGINS> file "ietf-system@2011-06-02.yang"

```
module ietf-system {
  namespace "urn:ietf:params:xml:ns:yang:ietf-system";
  prefix "sys";

  import ietf-inet-types {
    prefix inet;
  }

  import ietf-netconf-acm {
    prefix nacm;
  }

  import ietf-yang-types {
    prefix yang;
  }

  organization
    "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

  contact
    "WG Web:   <http://tools.ietf.org/wg/netmod/>
     WG List:  <mailto:netmod@ietf.org>

     WG Chair: David Kessens
               <mailto:david.kessens@nsn.com>

     WG Chair: Juergen Schoenwaelder
               <mailto:j.schoenwaelder@jacobs-university.de>

     Editor:   Andy Bierman
               <mailto:andy.bierman@brocade.com>

     Editor:   Martin Bjorklund
               <mailto:mbj@tail-f.com>";

  description
    "This module contains a collection of YANG definitions for the
     configuration and identification of the management system of a
```

```
       device.

       Copyright (c) 2011 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Simplified BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX; see
       the RFC itself for full legal notices.";

    // RFC Ed.: replace XXXX with actual RFC number and remove this
    // note.

    // RFC Ed.: update the date below with the date of RFC publication
    // and remove this note.
    revision 2011-06-02 {
      description
        "Initial revision.";
      reference
        "RFC XXXX: A YANG Data Model for System Management";
    }

    /*
     * Typedefs
     */

     typedef crypt-hash {
       type string {
         pattern "$0$.* | $1|5|6$[a-zA-Z0-9./]{2,16}$.*";
       }
       description
         "The crypt-hash type is used to store passwords using
          a hash function.  This type is implemented in various UNIX
          systems as the function crypt(3).

          When a clear text value is set to a leaf of this type, the
          server calculates a password hash, and stores the result
          in the datastore.  Thus, the password is never stored in
          clear text.

          When a leaf of this type is read, the stored password hash is
          returned.
```

```
          A value of this type matches one of the forms:

            $0$<clear text password>
            $<id>$<salt>$<password hash>

          The '$0$' prefix signals that the value is clear text.  When
          such a value is received by the server, a hash value is
          calculated, and the string '$<id>$<salt>$' is prepended to the
          result, where <salt> is a random 2-16 characters long salt
          used to generate the digest.  This value is stored in the
          configuration data store.

          If a value starting with '$<id>$<salt>$' is received, the
          server knows that the value already represents a hashed value,
          and stores it as is in the data store.

          When a server needs to verify a password given by a user, it
          finds the stored password hash string for that user, extracts
          the salt, and calculates the hash with the salt and given
          password as input.  If the calculated hash value is the same
          as the stored value, the password given by the client is
          correct.

          This type defines the following hash functions:

            id | hash function | feature
            ---+---------------+-------------------
             1 | MD5           | crypt-hash-md5
             5 | SHA-256       | crypt-hash-sha-256
             6 | SHA-512       | crypt-hash-sha-512

          The server indicates support for the different hash functions
          by advertising the corresponding feature.";
        reference
          "Wikipedia: http://en.wikipedia.org/wiki/Crypt_(Unix)
           RFC 1321: The MD5 Message-Digest Algorithm
           FIPS.180-3.2008: Secure Hash Standard";
      }

      /*
       * Features
       */

      feature authentication {
        description
          "Indicates that the device can be configured
           to do authentication of users.";
      }
```

```
    feature radius {
      if-feature authentication;
      description
        "Indicates that the device can be
         configured to act as a NAS and authenticate users
         with RADIUS.";
      reference
        "RFC 2865: Remote Authentication Dial In User Service (RADIUS)
         RFC 5607: Remote Authentication Dial-In User Service (RADIUS)
                   Authorization for Network Access Server (NAS)
                   Management";
    }

    feature local-users {
      if-feature authentication;
      description
        "Indicates that the device supports
         local user authentication.";
    }

    feature crypt-hash-md5 {
      description
        "Indicates that the device supports the MD5
         hash function in 'crypt-hash' values";
      reference "RFC 1321: The MD5 Message-Digest Algorithm";
    }

    feature crypt-hash-sha-256 {
      description
        "Indicates that the device supports the SHA-256
         hash function in 'crypt-hash' values";
      reference "FIPS.180-3.2008: Secure Hash Standard";
    }

    feature crypt-hash-sha-512 {
      description
        "Indicates that the device supports the SHA-512
         hash function in 'crypt-hash' values";
      reference "FIPS.180-3.2008: Secure Hash Standard";
    }

    feature ntp {
      description
        "Indicates that the device can be configured
         to use one or more NTP servers to set the
         system date and time.";
    }
```

```
    feature tz-database {
      description
        "Indicates that the local timezone on the device
         can be configured to use the TZ database
         to set the timezone and manage daylight savings time.";
      reference
        "TZ Database  http://www.twinsun.com/tz/tz-link.htm
         Maintaining the Timezone Database
         http://www.ietf.org/id/draft-lear-iana-timezone-database-04.txt
        ";
    }

    feature tz-enumeration {
      description
        "Indicates that the local timezone on the device
         can be configured using the timezone enumeration
         strings as an alias for an UTC offset.";
      reference
        "Wikipedia: http://en.wikipedia.org/wiki/"
      + "List_of_time_zone_abbreviations";
    }

    /*
     * Identities
     */

    identity authentication-method {
      description
        "Base identity for user authentication methods.";
    }

    identity radius {
      base authentication-method;
      description
        "Indicates user authentication using RADIUS.";
      reference
        "RFC 2865: Remote Authentication Dial In User Service (RADIUS)
         RFC 5607: Remote Authentication Dial-In User Service (RADIUS)
                   Authorization for Network Access Server (NAS)
                   Management";
    }

    identity local-users {
      base authentication-method;
      description
        "Indicates password-based authentication of locally
         configured users.";
    }
```

```
    /*
     * Top-level container
     */

    container system {
      description
        "System group configuration.";

      leaf contact {
        type string {
          length "0..255";
        }
        default "";
        reference
          "RFC 3418 - Management Information Base (MIB) for the
                      Simple Network Management Protocol (SNMP)
                      SNMPv2-MIB.sysContact";
      }

      leaf name {
        type string {
          length "0..255";
        }
        default "";
        reference
          "RFC 3418 - Management Information Base (MIB) for the
                      Simple Network Management Protocol (SNMP)
                      SNMPv2-MIB.sysName";
      }

      leaf location {
        type string {
          length "0..255";
        }
        default "";
        reference
          "RFC 3418 - Management Information Base (MIB) for the
                      Simple Network Management Protocol (SNMP)
                      SNMPv2-MIB.sysLocation";
      }

      container platform {
        description
          "Contains vendor-specific information for
           identifying the system platform and operating system.";
        reference
          "GNU coreutils homepage:
             http://www.gnu.org/software/coreutils
```

```
           Wikipedia: http://en.wikipedia.org/wiki/Uname";

        config false;

        leaf os-name {
          type string;
          description
            "The name of the operating system in use,
             for example 'Linux'";
          reference
            "uname --kernel-name";
        }

        leaf os-release {
          type string;
          description
            "The current release level of the operating
             system in use.  This string MAY indicate
             the OS source code revision.";
          reference
            "uname --kernel-release";
        }

        leaf os-version {
          type string;
          description
            "The current version level of the operating
             system in use.  This string MAY indicate
             the specific OS build date and target variant
             information.";
          reference
            "uname --kernel-version";
        }

        leaf machine {
          type string;
          description
            "A vendor-specific identifier string representing
             the hardware in use.";
          reference
            "uname --machine";
        }

        leaf nodename {
          type string;
          description
            "The host name of this system.";
          reference
```

```
          "uname --nodename";
      }
    }

    container clock {
      description
        "Configuration and monitoring of the system
         date and time properties.";

      leaf current-datetime {
        description
          "The current system date and time.";
        type yang:date-and-time;
        config false;
      }

      leaf boot-datetime {
        description
          "The system date and time when the NETCONF
           server last restarted.";
        type yang:date-and-time;
        config false;
      }

      choice timezone-info {
        description
          "Configure the system timezone information.";

        leaf tz-database-id {
          if-feature tz-database;
          description
            "The TZ database location identifier string
             to use for the system, such as 'Europe/Stockholm'.";
          type string;
        }

        leaf tz-enumeration-id {
          if-feature tz-enumeration;
          description
            "The timezone enumeration string to use
             for the system, such as 'CET'.";
          type string;
          // FIXME: use TimezoneEnum typedef instead
          // see http://en.wikipedia.org/wiki/
          //           List_of_time_zone_abbreviations
        }

        leaf utc-offset {
```

```
            description
              "The number of minutes to add to UTC time to
               identify the timezone for this system.
               For example, 'UTC - 8:00 hours' would be
               represented as '-480'.";
            type int16 {
              range "-1439 .. 1439";
            }
          }
        }
      }

      container ntp {
        if-feature ntp;

        description
          "Configuration of the NTP client.";

        leaf use-ntp {
          description
            "Indicates that the system should attempt
             to synchronize the system clock with an
             NTP server from the 'ntp-server' list.";
          type boolean;
          default true;
        }

        list ntp-server {
          description
            "List of NTP servers to use for
             system clock synchronization.  If 'use-ntp'
             is 'true', then the system will attempt to
             contact and utilize the specified NTP servers.";

          key address;

          leaf address {
            description
              "The IP address or domain name of the NTP server.";
            type inet:host;
          }

          // TBD: add more parameters here
          // and/or vendors can add parameters via augment
        }
      }

      container dns {
```

```
        description
          "Configuration of the DNS resolver.

           The 'domain' keyword of /etc/resolv.conf is not supported,
           since it is equivalent to 'search' with a single domain.";

        leaf-list search {
          type inet:host;
          ordered-by user;
        }
        leaf-list server {
          type inet:ip-address;
          ordered-by user;
          description
            "Addresses of the name servers that the resolver should
             query.

             Implementations MAY limit the number of entries in this
             leaf list.";
        }
        container options {
          description
            "Resolver options.  The set of available options has been
             limited to those that are generally available across
             different resolver implementations, and generally useful.";
          leaf ndots {
            type uint8;
            default "1";
          }
          leaf timeout {
            type uint8;
            units "seconds";
            default "5";
          }
          leaf attempts {
            type uint8;
            default "2";
          }
        }
      }

      container authentication {
        nacm:secure;
        if-feature authentication;

         description
           "The authentication configuration subtree.";
```

```
        leaf-list user-authentication-order {
          type identityref {
            base authentication-method;
          }
          must '(. = "sys:radius" and ../radius/server) or'
             + '(. != "sys:radius")' {
            error-message
              "When 'radius' is used, a radius server
               must be configured.";
          }
          ordered-by user;

          description
            "When the device authenticates a user with
             a password, it tries the authentication methods in this
             leaf-list in order.  If authentication with one method
             fails, the next method is used.  If no method succeeds,
             the user is denied access.

             If the 'radius' feature is advertised by the NETCONF
             server, the 'radius' identity can be added to this
             list.

             If the 'local-users' feature is advertised by the
             NETCONF server, the 'local-users' identity can be
             added to this list.";
        }

        container radius {
          if-feature radius;

          description
            "The RADIUS configuration for authentication.";

          list server {
            key address;
            ordered-by user;

            description
              "The RADIUS server configuration used by
               the device.";

            leaf address {
              type inet:host;
              description
                "The address of the RADIUS server.";
            }
            leaf port {
```

```
            type inet:port-number;
            default "1812";
            description
             "The port number of the RADIUS server.";
          }
          leaf shared-secret {
            type string;
            nacm:very-secure;
            description
              "The shared secret which is known to both the RADIUS
               client and server.";
            reference
              "RFC 2865: Remote Authentication Dial In User Service";
          }
        }
        container options {
          description
            "RADIUS client options.";

          leaf timeout {
            type uint8;
            units "seconds";
            default "5";
            description
              "The number of seconds the device will wait for a
               response from a RADIUS server before trying with a
               different server.";
          }
          leaf attempts {
            type uint8;
            default "2";
            description
              "The number of times the device will send a query to
               the RADIUS servers before giving up.";
          }
        }
      }

      list user {
        if-feature local-users;
        key name;

        description
          "The list of local users configured on this device.";

        leaf name {
          type string;
          description
```

```
              "The user name string identifying this entry.";
            }
            leaf password {
              type crypt-hash;
              description
                "The password for this entry.";
            }
            leaf ssh-dsa {
              type binary;
              description
                "The public DSA key for this entry.";
            }
            leaf ssh-rsa {
              type binary;
              description
                "The public RSA key for this entry.";
            }
          }
        }
      }

      rpc set-current-datetime {
        nacm:secure;
        description
          "Manually set the /system/clock/current-datetime leaf
           to the specified value.

           If the /system/ntp/ntp-in-use leaf exists and
           is set to 'true', then this operation will
           fail with error-tag 'operation-failed',
           and error-app-tag value of 'ntp-active'";
        input {
          leaf current-datetime {
            description
              "The current system date and time.";
            type yang:date-and-time;
            mandatory true;
          }
        }
      }


    }


     <CODE ENDS>
```

5. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in RFC 3688, the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-system

Registrant Contact: The NETMOD WG of the IETF.

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

This document registers a YANG module in the YANG Module Names registry [RFC6020].

```
name:         ietf-system
namespace:    urn:ietf:params:xml:ns:yang:ietf-system
prefix:       sys
reference:    RFC XXXX
```

6. Security Considerations

TBD.

7. Normative References

[I-D.ietf-netconf-rfc4742bis]
Wasserman, M. and T. Goddard, "Using the NETCONF Configuration Protocol over Secure Shell (SSH)", draft-ietf-netconf-rfc4742bis-08 (work in progress), March 2011.

[I-D.lear-iana-timezone-database]
Lear, E. and P. Eggert, "IANA Procedures for Maintaining the Timezone Database", draft-lear-iana-timezone-database-04 (work in progress), May 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4252] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Authentication Protocol", RFC 4252, January 2006.

[RFC4253] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, January 2006.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

Authors' Addresses

Andy Bierman
Brocade

Email: andy.bierman@brocade.com

Martin Bjorklund
Tail-f Systems

Email: mbj@tail-f.com