Network Configuration Access Control Model

Abstract

   The standardization of network configuration interfaces for use with
   the Network Configuration Protocol (NETCONF) or the RESTCONF protocol
   requires a structured and secure operating environment that promotes
   human usability and multi-vendor interoperability. There is a need
   for standard mechanisms to restrict NETCONF or RESTCONF protocol
   access for particular users to a preconfigured subset of all
   available NETCONF or RESTCONF protocol operations and content. This
   document defines such an access control model.

   This document obsoletes RFC 6536.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 7841.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   https://www.rfc-editor.org/info/rfc8341.

Copyright Notice

Table of Contents

1. Introduction

   The Network Configuration Protocol (NETCONF) and the RESTCONF
   protocol do not provide any standard mechanisms to restrict the
   protocol operations and content that each user is authorized to
   access.

   There is a need for interoperable management of the controlled access
   to administrator-selected portions of the available NETCONF or
   RESTCONF content within a particular server.

   This document addresses access control mechanisms for the Operations
   and Content layers of NETCONF, as defined in [RFC6241]; and RESTCONF,
   as defined in [RFC8040]. It contains three main sections:

   1. Access Control Design Objectives

   2. NETCONF Access Control Model (NACM)

   3. YANG Data Model (ietf-netconf-acm.yang)

   YANG version 1.1 [RFC7950] adds two new constructs that need special
   access control handling. The "action" statement is similar to the
   "rpc" statement, except that it is located within a data node. The
   "notification" statement can also be located within a data node.

1.1. Terminology

   The key words "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] [RFC8174] when, and only when, they appear in all
   capitals, as shown here.

   The following terms are defined in [RFC8342] and are not redefined
   here:

   o datastore

   o configuration datastore

   o conventional configuration datastore

   o candidate configuration datastore

   o running configuration datastore

   o startup configuration datastore

   o operational state datastore

   o client

   o server

   The following terms are defined in [RFC6241] and are not redefined
   here:

   o protocol operation

   o session

   o user

   The following terms are defined in [RFC7950] and are not redefined
   here:

   o action

   o data node

   o data definition statement

   The following terms are defined in [RFC8040] and are not redefined
   here:

   o data resource

   o datastore resource

   o operation resource

   o target resource

   The following term is defined in [RFC7230] and is not redefined here:

   o request URI

   The following terms are used throughout this document:

   access control: A security feature provided by the server that
      allows an administrator to restrict access to a subset of all
      protocol operations and data, based on various criteria.

   access control model (ACM): A conceptual model used to configure and
      monitor the access control procedures desired by the administrator
      to enforce a particular access control policy.

   access control rule: The criterion used to determine if a particular
      access operation will be permitted or denied.

   access operation: How a request attempts to access a conceptual
      object. One of "none", "read", "create", "delete", "update", or
      "execute".

   data node hierarchy: The hierarchy of data nodes that identifies the
      specific "action" or "notification" node in the datastore.

   recovery session: A special administrative session that is given
      unlimited NETCONF access and is exempt from all access control
      enforcement. The mechanism or mechanisms used by a server to
      control and identify whether or not a session is a recovery
      session are implementation specific and are outside the scope of
      this document.

   write access: A shorthand for the "create", "delete", and "update"
      access operations.

1.2. Changes since RFC 6536

   The NACM procedures and data model have been updated to support new
   data modeling capabilities in version 1.1 of the YANG data modeling
   language. The "action" and "notification" statements can be used
   within data nodes to define data-model-specific operations and
   notifications.

   An important use case for these new YANG statements is the increased
   access control granularity that can be achieved over top-level "rpc"
   and "notification" statements. The new "action" and "notification"
   statements are used within data nodes, and access to the action or
   notification can be restricted to specific instances of these data
   nodes.

   Support for the RESTCONF protocol has been added. The RESTCONF
   operations are similar to the NETCONF operations, so a simple mapping
   to the existing NACM procedures and data model is possible.

   The data node access behavior for path matches has been clarified to
   also include matching descendant nodes of the specified path.

   The <edit-config> operation access rights behavior has been clarified
   to indicate that write access is not required for data nodes that are
   implicitly modified through side effects (such as the evaluation of
   YANG when-stmts, or data nodes implicitly deleted when creating a
   data node under a different branch under a YANG choice-stmt).

The Security Considerations section has been updated to comply with
the "YANG module security guidelines" [YANG-SEC]. Note that the YANG
module in this document does not define any RPC operations.

2. Access Control Design Objectives

This section documents the design objectives for the NETCONF access
control model presented in Section 3.

2.1. Access Control Points

NETCONF allows server implementers to add new custom protocol
operations, and the YANG data modeling language supports this
feature. These operations can be defined in standard or proprietary
YANG modules.

It is not possible to design an ACM for NETCONF that only focuses on
a static set of standard protocol operations defined by NETCONF
itself, like some other protocols. Since few assumptions can be made
about an arbitrary protocol operation, the NETCONF architectural
server components need to be protected at three conceptual control
points.

These access control points, described in Figure 1, are as follows:

protocol operation: Permission to invoke specific protocol
   operations.

datastore: Permission to read and/or alter specific data nodes
   within any datastore.

notification: Permission to receive specific notification event
   types.

```
               +-------------+                  +-------------+
  client       |  protocol   |                  |  data node  |
  request -->  |  operation  | ------------->   |   access    |
               |  allowed?   |   datastore      |  allowed?   |
               +-------------+   or state       +-------------+
                                 data access


               +----------------+
               |  notification  |
  event -->    |  allowed?      |
               +----------------+
```
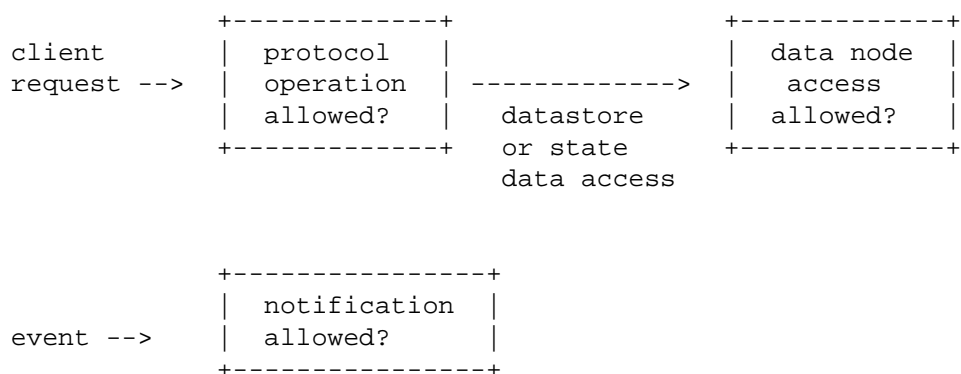
Figure 1

2.2. Simplicity

   There is concern that a complicated ACM will not be widely deployed
   because it is too hard to use. Configuration of the access control
   system needs to be as simple as possible. Simple and common tasks
   need to be easy to configure and require little expertise or
   domain-specific knowledge. Complex tasks are possible using
   additional mechanisms that may require additional expertise.

   A single set of access control rules ought to be able to control all
   types of NETCONF protocol operation invocation, all datastore access,
   and all notification events.

   Access control ought to be defined with a small and familiar set of
   permissions, while still allowing full control of datastore access.

2.3. Procedural Interface

   NETCONF uses a Remote Procedure Call (RPC) model and an extensible
   set of protocol operations. Access control for any possible protocol
   operation is necessary.

2.4. Datastore Access

   It is necessary to control access to specific nodes and subtrees
   within the datastore, regardless of which protocol operation --
   standard or proprietary -- was used to access the datastore.

2.5. Users and Groups

   It is necessary that access control rules for a single user or a
   configurable group of users can be configured.

   The ACM needs to support the concept of administrative groups, to
   support the well-established distinction between a root account and
   other types of less-privileged conceptual user accounts. These
   groups need to be configurable by the administrator.

   It is necessary that the user-to-group mapping can be delegated to a
   central server, such as a RADIUS server [RFC2865] [RFC5607]. Since
   authentication is performed by the transport layer and RADIUS
   performs authentication and service authorization at the same time,
   the underlying transport protocol needs to be able to report a set of
   group names associated with the user to the server. It is necessary
   that the administrator can disable the usage of these group names
   within the ACM.

2.6. Maintenance

   It ought to be possible to disable part or all of the access control
   model enforcement procedures without deleting any access control
   rules.

2.7. Configuration Capabilities

   Suitable configuration and monitoring mechanisms are needed to allow
   an administrator to easily manage all aspects of the ACM's behavior.
   A standard data model, suitable for use with the <edit-config>
   protocol operation, needs to be available for this purpose.

   Access control rules to restrict access operations on specific
   subtrees within the configuration datastore need to be supported.

2.8. Identifying Security-Sensitive Content

   One of the most important aspects of the data model documentation,
   and one of the biggest concerns during deployment, is the
   identification of security-sensitive content. This applies to
   protocol operations in NETCONF, not just data and notifications.

   It is mandatory for security-sensitive objects to be documented in
   the Security Considerations section of an RFC. This is nice, but it
   is not good enough, for the following reasons:

   o This documentation-only approach forces administrators to study
      the RFC and determine if there are any potential security risks
      introduced by a new data model.

   o If any security risks are identified, then the administrator must
      study some more RFC text and determine how to mitigate the
      security risk(s).

   o The ACM on each server must be configured to mitigate the security
      risks, e.g., require privileged access to read or write the
      specific data identified in the Security Considerations section.

   o If the ACM is not preconfigured, then there will be a time window
      of vulnerability after the new data model is loaded and before the
      new access control rules for that data model are configured,
      enabled, and debugged.

   Often, the administrator just wants to disable default access to the
   secure content so that no inadvertent or malicious changes can be
   made to the server. This allows the default rules to be more
   lenient, without significantly increasing the security risk.

A data model designer needs to be able to use machine-readable
statements to identify content that needs to be protected by default.
This will allow client and server tools to automatically identify
data-model-specific security risks, by denying access to sensitive
data unless the user is explicitly authorized to perform the
requested access operation.

3. NETCONF Access Control Model (NACM)

3.1. Overview

This section provides a high-level overview of the access control
model structure. It describes the NETCONF protocol message
processing model and the conceptual access control requirements
within that model.

3.1.1. Features

The NACM data model provides the following features:

o Independent control of RPC, action, data, and notification access
   is provided.

o The concept of an emergency recovery session is supported, but
   configuration of the server for this purpose is beyond the scope
   of this document. An emergency recovery session will bypass all
   access control enforcement, in order to allow it to initialize or
   repair the NACM configuration.

o A simple and familiar set of datastore permissions is used.

o Support for YANG security tagging (e.g., a
   "nacm:default-deny-write" statement) allows default security modes
   to automatically exclude sensitive data.

o Separate default access modes for read, write, and execute
   permissions are provided.

o Access control rules are applied to configurable groups of users.

o The access control enforcement procedures can be disabled during
   operation, without deleting any access control rules, in order to
   debug operational problems.

   o The number of denied protocol operation requests and denied
      datastore write requests can be monitored by the client.

   o Simple unconstrained YANG instance-identifiers are used to
      configure access control rules for specific data nodes.

3.1.2. External Dependencies

   NETCONF [RFC6241] and RESTCONF [RFC8040] are used for network
   management purposes within this document.

   The YANG data modeling language [RFC7950] is used to define the data
   models for use with NETCONF or RESTCONF. YANG is also used to define
   the data model in this document.

3.1.3. Message Processing Model

   The following diagram shows the conceptual message flow model,
   including the points at which access control is applied during
   NETCONF message processing.

   RESTCONF operations are mapped to the access control model based on
   the HTTP method and resource class used in the operation. For
   example, a POST method on a data resource is considered "write data
   node" access, but a POST method on an operation resource is
   considered "operation" access.

   The new "pre-read data node acc. ctl" boxes in the diagram below
   refer to group read access as it relates to data node ancestors of an
   action or notification. As an example, if an action is defined as
   /interfaces/interface/reset-interface, the group must be authorized
   to (1) read /interfaces and /interfaces/interface and (2) execute on
   /interfaces/interface/reset-interface.

```
                 +-------------------------+
                 |         session         |
                 |        (username)       |
                 +-------------------------+
                      |               ^
                      V               |
              +--------------+   +---------------+
              |   message    |   |   message     |
              |  dispatcher  |   |  generator    |
              +--------------+   +---------------+
                   |   |           ^         ^
                   |   V           |         |
                   | +=============+         |
                   | | pre-read    |         |
                   | | data node   |         |
                   | | acc. ctl    |         |
                   | +=============+         |
                   | |                       |
                   V V                       |
            +===========+  +-------------+  +----------------+
            | operation |--->|   reply   |  | <notification> |
            | acc. ctl  |  |  generator  |  |   generator    |
            +===========+  +-------------+  +----------------+
                  |          ^     ^              ^
                  V      +------+   |              |
            +-----------+ |  +=============+  +================+
            | operation | |  |    read     |  | <notification> |
            | processor |-+  | data node   |  |   access ctl   |
            |           | |  | acc. ctl    |  |                |
            +-----------+ |  +=============+  +================+
                 | |         ^          ^   ^   ^
                 V +---------------+ |      |   |
            +===========+         | |      | +===========+
            |  write    |         | |      | | pre-read  |
            | data node |         | |      | | data node |
            | acc. ctl  | ----------+ |      | | acc. ctl  |
            +===========+         | | |      | +===========+
                 |                | | |      |   ^
                 V                V V |      |   |
            +--------------+  +------------------+
            | configuration| --->|     server      |
            |  datastore   |  |  instrumentation   |
            |              | <---|                 |
            +--------------+  +------------------+
```

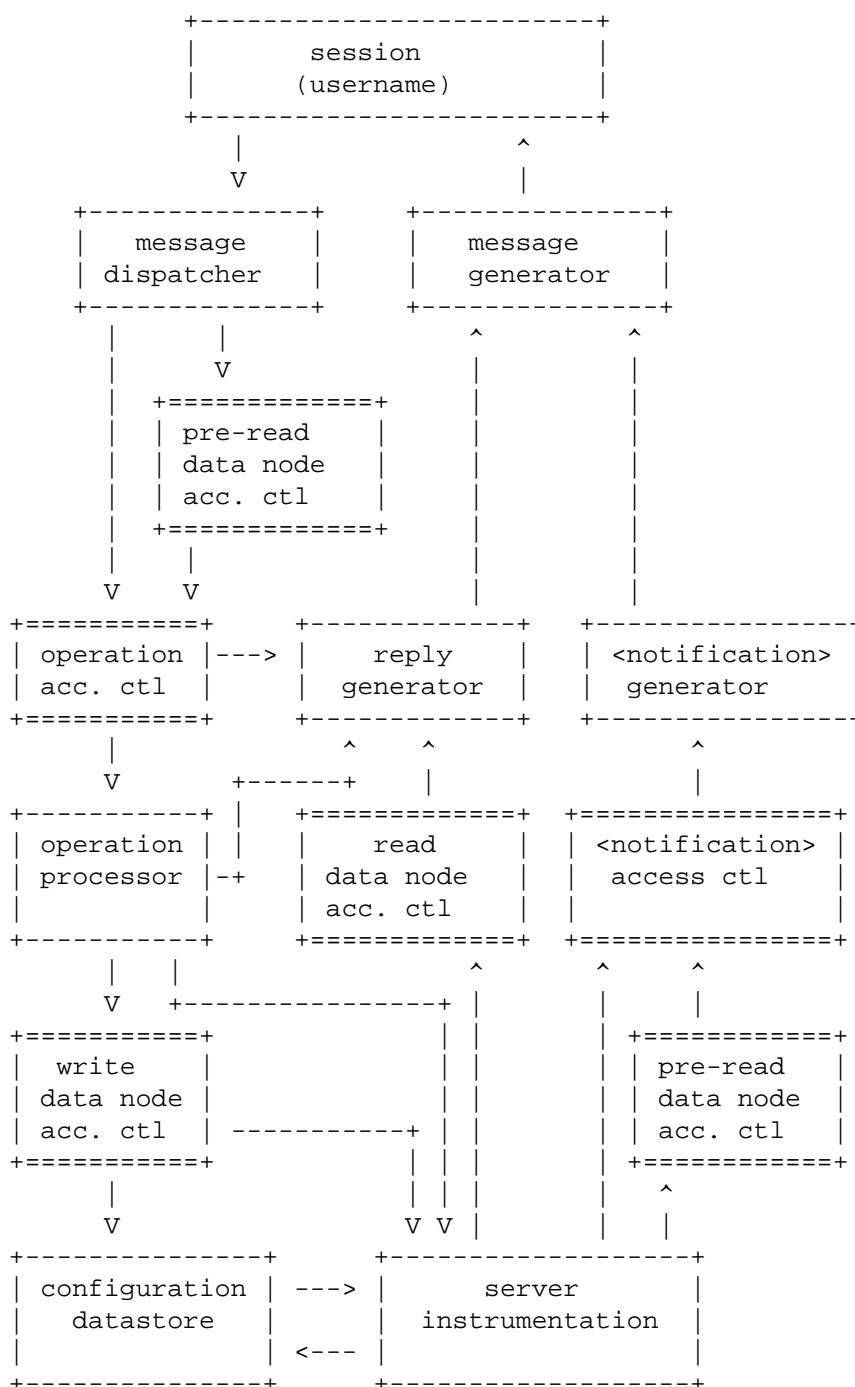   Figure 2

The following high-level sequence of conceptual processing steps is
executed for each received <rpc> message, if access control
enforcement is enabled:

o For each active session, access control is applied individually to
   all <rpc> messages (except <close-session>) received by the
   server, unless the session is identified as a recovery session.

o If the <action> operation defined in [RFC7950] is invoked, then
   read access is required for all instances in the hierarchy of data
   nodes that identifies the specific action in the datastore, and
   execute access is required for the action node. If the user is
   not authorized to read all the specified data nodes and execute
   the action, then the request is rejected with an "access-denied"
   error.

o Otherwise, if the user is not authorized to execute the specified
   protocol operation, then the request is rejected with an
   "access-denied" error.

o If a datastore is accessed by the protocol operation, then the
   server checks to see if the client is authorized to access the
   nodes in the datastore. If the user is not authorized to perform
   the requested access operation on the requested data, then the
   request is rejected with an "access-denied" error.

The following sequence of conceptual processing steps is executed for
each generated notification event, if access control enforcement is
enabled:

o Server instrumentation generates a notification for a particular
   subscription.

o If the "notification" statement is specified within a data
   subtree, as specified in [RFC7950], then read access is required
   for all instances in the hierarchy of data nodes that identifies
   the specific notification in the datastore, and read access is
   required for the notification node. If the user is not authorized
   to read all the specified data nodes and the notification node,
   then the notification is dropped for that subscription.

o If the "notification" statement is a top-level statement, the
   notification access control enforcer checks the notification event
   type, and if it is one that the user is not authorized to read,
   then the notification is dropped for that subscription.

3.2. Datastore Access

   The same access control rules apply to all datastores that support
   the NACM -- for example, the candidate configuration datastore or the
   running configuration datastore.

   All conventional configuration datastores and the operational state
   datastore are controlled by the NACM. Local files, remote files, or
   datastores accessed via the <url> parameter are not controlled by
   the NACM.

3.2.1. Mapping New Datastores to NACM

   It is possible that new datastores will be defined over time for use
   with NETCONF. The NACM MAY be applied to other datastores that have
   similar access rights as defined in the NACM. To apply the NACM to a
   new datastore, the new datastore specification needs to define how it
   maps to the NACM CRUDX (Create, Read, Update, Delete, eXec) access
   rights. It is possible that only a subset of the NACM access rights
   would be applicable. For example, only retrieval access control
   would be needed for a read-only datastore. Operations and access
   rights not supported by the NACM CRUDX model are outside the scope of
   this document. A datastore does not need to use the NACM, e.g., the
   datastore specification defines something else or does not use access
   control.

3.2.2. Access Rights

   A small set of hard-wired datastore access rights is needed to
   control access to all possible protocol operations, including vendor
   extensions to the standard protocol operation set.

   The CRUDX model can support all protocol operations:

   o Create: allows the client to add a new data node instance to a
      datastore.

   o Read: allows the client to read a data node instance from a
      datastore or receive the notification event type.

   o Update: allows the client to update an existing data node instance
      in a datastore.

   o Delete: allows the client to delete a data node instance from a
      datastore.

   o eXec: allows the client to execute the operation.

3.2.3. RESTCONF Methods

   The RESTCONF protocol utilizes HTTP methods to perform datastore
   operations, similar to NETCONF. The NACM procedures were originally
   written for NETCONF protocol operations, so the RESTCONF methods are
   mapped to NETCONF operations for the purpose of access control
   processing. The enforcement procedures described within this
   document apply to both protocols unless explicitly stated otherwise.

   The request URI needs to be considered when processing RESTCONF
   requests on data resources:

   o For HEAD and GET requests, any data nodes that are ancestor nodes
      of the target resource are considered to be part of the retrieval
      request for access control purposes.

   o For PUT, PATCH, and DELETE requests, any data nodes that are
      ancestor nodes of the target resource are not considered to be
      part of the edit request for access control purposes. The access
      operation for these nodes is considered to be "none". The edit
      begins at the target resource.

   o For POST requests on data resources, any data nodes that are
      specified in the request URI, including the target resource, are
      not considered to be part of the edit request for access control
      purposes. The access operation for these nodes is considered to
      be "none". The edit begins at a child node of the target
      resource, specified in the message body.

Not all RESTCONF methods are subject to access control. The
following table specifies how each method is mapped to NETCONF
protocol operations. The value "none" indicates that the NACM is not
applied at all to the specific RESTCONF method.

```
+---------+----------------+--------------------+----------------+
| Method  | Resource class | NETCONF operation  | Access         |
|         |                |                    | operation      |
+---------+----------------+--------------------+----------------+
| OPTIONS | all            | none               | none           |
| HEAD    | all            | <get>, <get-config> | read          |
| GET     | all            | <get>, <get-config> | read          |
| POST    | datastore, data | <edit-config>     | create         |
| POST    | operation      | specified operation | execute       |
| PUT     | data           | <edit-config>      | create, update |
| PUT     | datastore      | <copy-config>      | update         |
| PATCH   | data, datastore | <edit-config>     | update         |
| DELETE  | data           | <edit-config>      | delete         |
+---------+----------------+--------------------+----------------+
```

Table 1: Mapping RESTCONF Methods to NETCONF

3.2.4. <get> and <get-config> Operations

The NACM access rights are not directly coupled to the <get> and
<get-config> protocol operations but apply to all <rpc> operations
that would result in a "read" access operation to the target
datastore. This section describes how these access rights apply to
the specific access operations supported by the <get> and
<get-config> protocol operations.

Data nodes to which the client does not have read access are silently
omitted, along with any descendants, from the <rpc-reply> message.
This is done to allow NETCONF filters for <get> and <get-config> to
function properly, instead of causing an "access-denied" error
because the filter criteria would otherwise include unauthorized read
access to some data nodes. For NETCONF filtering purposes, the
selection criteria are applied to the subset of nodes that the user
is authorized to read, not the entire datastore.

3.2.5. <edit-config> Operation

The NACM access rights are not directly coupled to the <edit-config>
"operation" attribute, although they are similar. Instead, a NACM
access right applies to all protocol operations that would result in
a particular access operation to the target datastore. This section
describes how these access rights apply to the specific access
operations supported by the <edit-config> protocol operation.

If the effective access operation is "none" (i.e.,
default-operation="none") for a particular data node, then no access
control is applied to that data node. This is required to allow
access to a subtree within a larger data structure. For example, a
user may be authorized to create a new "/interfaces/interface" list
entry but not be authorized to create or delete its parent container
("/interfaces"). If the "/interfaces" container already exists in
the target datastore, then the effective operation will be "none" for
the "/interfaces" node if an "/interfaces/interface" list entry is
edited.

If the protocol operation would result in the creation of a datastore
node and the user does not have "create" access permission for that
node, the protocol operation is rejected with an "access-denied"
error.

If the protocol operation would result in the deletion of a datastore
node and the user does not have "delete" access permission for that
node, the protocol operation is rejected with an "access-denied"
error.

If the protocol operation would result in the modification of a
datastore node and the user does not have "update" access permission
for that node, the protocol operation is rejected with an
"access-denied" error.

A "merge" or "replace" <edit-config> operation may include data nodes
that do not alter portions of the existing datastore. For example, a
container or list node may be present for naming purposes but does
not actually alter the corresponding datastore node. These unaltered
data nodes are ignored by the server and do not require any access
rights by the client.

A "merge" <edit-config> operation may include data nodes but not
include particular child data nodes that are present in the
datastore. These missing data nodes within the scope of a "merge"
<edit-config> operation are ignored by the server and do not require
any access rights by the client.

The contents of specific restricted datastore nodes MUST NOT be
exposed in any <rpc-error> elements within the reply.

An <edit-config> operation may cause data nodes to be implicitly
created or deleted as an implicit side effect of a requested
operation. For example, a YANG when-stmt expression may evaluate to
a different result, causing data nodes to be deleted, or created with
default values; or if a data node is created under one branch of a
YANG choice-stmt, then all data nodes under the other branches are

implicitly removed. No NACM access rights are required on any data
nodes that are implicitly changed as a side effect of another allowed
operation.

3.2.6. <copy-config> Operation

   Access control for the <copy-config> protocol operation requires
   special consideration because the administrator may be replacing the
   entire target datastore.

   If the source of the <copy-config> protocol operation is the running
   configuration datastore and the target is the startup configuration
   datastore, the client is only required to have permission to execute
   the <copy-config> protocol operation.

   Otherwise:

   o If the source of the <copy-config> operation is a datastore, then
      data nodes to which the client does not have read access are
      silently omitted.

   o If the target of the <copy-config> operation is a datastore, the
      client needs access to the modified nodes. Specifically:

      * If the protocol operation would result in the creation of a
         datastore node and the user does not have "create" access
         permission for that node, the protocol operation is rejected
         with an "access-denied" error.

      * If the protocol operation would result in the deletion of a
         datastore node and the user does not have "delete" access
         permission for that node, the protocol operation is rejected
         with an "access-denied" error.

      * If the protocol operation would result in the modification of a
         datastore node and the user does not have "update" access
         permission for that node, the protocol operation is rejected
         with an "access-denied" error.

3.2.7. <delete-config> Operation

   Access to the <delete-config> protocol operation is denied by
   default. The "exec-default" leaf does not apply to this protocol
   operation. Access control rules must be explicitly configured to
   allow invocation by a non-recovery session.

3.2.8. <commit> Operation

   The server MUST determine the exact nodes in the running
   configuration datastore that are actually different and only check
   "create", "update", and "delete" access permissions for this set of
   nodes, which could be empty.

   For example, if a session can read the entire datastore but only
   change one leaf, that session needs to be able to edit and commit
   that one leaf.

3.2.9. <discard-changes> Operation

   The client is only required to have permission to execute the
   <discard-changes> protocol operation. No datastore permissions are
   needed.

3.2.10. <kill-session> Operation

   The <kill-session> operation does not directly alter a datastore.
   However, it allows one session to disrupt another session that is
   editing a datastore.

   Access to the <kill-session> protocol operation is denied by default.
   The "exec-default" leaf does not apply to this protocol operation.
   Access control rules must be explicitly configured to allow
   invocation by a non-recovery session.

3.3. Model Components

   This section defines the conceptual components related to the access
   control model.

3.3.1. Users

   A "user" is the conceptual entity that is associated with the access
   permissions granted to a particular session. A user is identified by
   a string that is unique within the server.

   As described in [RFC6241], the username string is derived from the
   transport layer during session establishment. If the transport layer
   cannot authenticate the user, the session is terminated.

3.3.2. Groups

Access to a specific NETCONF protocol operation is granted to a
session. The session is associated with a group (i.e., not with
a user).

A group is identified by its name. All group names are unique within
the server.

Access control is applied at the level of groups. A group contains
zero or more group members.

A group member is identified by a username string.

The same user can be a member of multiple groups.

3.3.3. Emergency Recovery Session

The server MAY support a recovery session mechanism, which will
bypass all access control enforcement. This is useful for
restricting initial access and repairing a broken access control
configuration.

3.3.4. Global Enforcement Controls

There are five global controls that are used to help control how
access control is enforced.

3.3.4.1. enable-nacm Switch

A global "enable-nacm" on/off switch is provided to enable or disable
all access control enforcement. When this global switch is set to
"true", all requests are checked against the access control rules and
only permitted if configured to allow the specific access request.
When this global switch is set to "false", all access requests are
permitted.

3.3.4.2. read-default Switch

An on/off "read-default" switch is provided to enable or disable
default access to receive data in replies and notifications. When
the "enable-nacm" global switch is set to "true", this global switch
is relevant if no matching access control rule is found to explicitly
permit or deny read access to the requested datastore data or
notification event type.

When this global switch is set to "permit" and no matching access
control rule is found for the datastore read or notification event
requested, access is permitted.

When this global switch is set to "deny" and no matching access
control rule is found for the datastore read or notification event
requested, access is denied. This means that the requested data is
not sent to the client. See step 11 in Section 3.4.5 for details.

3.3.4.3. write-default Switch

An on/off "write-default" switch is provided to enable or disable
default access to alter configuration data. When the "enable-nacm"
global switch is set to "true", this global switch is relevant if no
matching access control rule is found to explicitly permit or deny
write access to the requested datastore data.

When this global switch is set to "permit" and no matching access
control rule is found for the datastore write requested, access is
permitted.

When this global switch is set to "deny" and no matching access
control rule is found for the datastore write requested, access is
denied. See step 12 in Section 3.4.5 for details.

3.3.4.4. exec-default Switch

An on/off "exec-default" switch is provided to enable or disable
default access to execute protocol operations. When the
"enable-nacm" global switch is set to "true", this global switch is
relevant if no matching access control rule is found to explicitly
permit or deny access to the requested NETCONF protocol operation.

When this global switch is set to "permit" and no matching access
control rule is found for the NETCONF protocol operation requested,
access is permitted.

When this global switch is set to "deny" and no matching access
control rule is found for the NETCONF protocol operation requested,
access is denied. See step 12 in Section 3.4.4 and step 13 in
Section 3.4.5 for details.

3.3.4.5. enable-external-groups Switch

   When this global switch is set to "true", the group names reported by
   the transport layer for a session are used together with the locally
   configured group names to determine the access control rules for the
   session.

   When this switch is set to "false", the group names reported by the
   transport layer are ignored by the NACM.

3.3.5. Access Control Rules

   There are four types of rules available in the NACM:

   module rule: controls access for definitions in a specific YANG
      module, identified by its name.

   protocol operation rule: controls access for a specific protocol
      operation, identified by its YANG module and name.

   data node rule: controls access for a specific data node and its
      descendants, identified by its path location within the conceptual
      XML document for the data node.

   notification rule: controls access for a specific notification event
      type, identified by its YANG module and name.

3.4. Access Control Enforcement Procedures

   There are six separate phases that need to be addressed, four of
   which are related to the NETCONF message processing model
   (Section 3.1.3):

   1. Initial operation

   2. Session establishment

   3. "access-denied" error handling

   4. Incoming RPC message validation

   5. Data node access validation

   6. Outgoing <notification> authorization

   In addition, the initial startup mode for a NETCONF server, session
   establishment, and "access-denied" error-handling procedures also
   need to be considered.

The server MUST use the access control rules in effect at the time it
starts processing the message. The same access control rules MUST
stay in effect for the processing of the entire message.

3.4.1. Initial Operation

Upon the very first startup of the NETCONF server, the access control
configuration will probably not be present. If it isn't, a server
MUST NOT allow any write access to any session role except a recovery
session.

Access rules are enforced any time a request is initiated from a user
session. Access control is not enforced for server-initiated access
requests, such as the initial load of the running configuration
datastore, during bootup.

3.4.2. Session Establishment

The access control model applies specifically to the well-formed XML
content transferred between a client and a server after session
establishment has been completed and after the <hello> exchange has
been successfully completed.

Once session establishment is completed and a user has been
authenticated, the transport layer reports the username and a
possibly empty set of group names associated with the user to the
NETCONF server. The NETCONF server will enforce the access control
rules, based on the supplied username, group names, and the
configuration data stored on the server.

3.4.3. "access-denied" Error Handling

The "access-denied" error-tag is generated when the access control
system denies access to either a request to invoke a protocol
operation or a request to perform a particular access operation on
the configuration datastore.

A server MUST NOT include any information the client is not allowed
to read in any <error-info> elements within the <rpc-error> response.

3.4.4. Incoming RPC Message Validation

   The diagram below shows the basic conceptual structure of the access
   control processing model for incoming NETCONF <rpc> messages within a
   server.

```
                NETCONF server
                +------------+
                |    XML     |
                |  message   |
                | dispatcher |
                +------------+
                      |
                      |
                      V
             +---------------+
             | <rpc> message |
             +---------------+
                 |    |    |
                 |    |    +------------------------------+
                 |    +--------------+                    |
                 V                   V                    V
      +-----------------+ +-------------------+ +-------------------+
      | vendor operation | | standard operation | | standard operation |
      |    <my-edit>     | |   <edit-config>   | |     <unlock>      |
      +-----------------+ +-------------------+ +-------------------+
                 |                   |
                 |                   |
                 V                   V
             +---------------------+
             |    configuration    |
             |     datastore       |
             +---------------------+
```
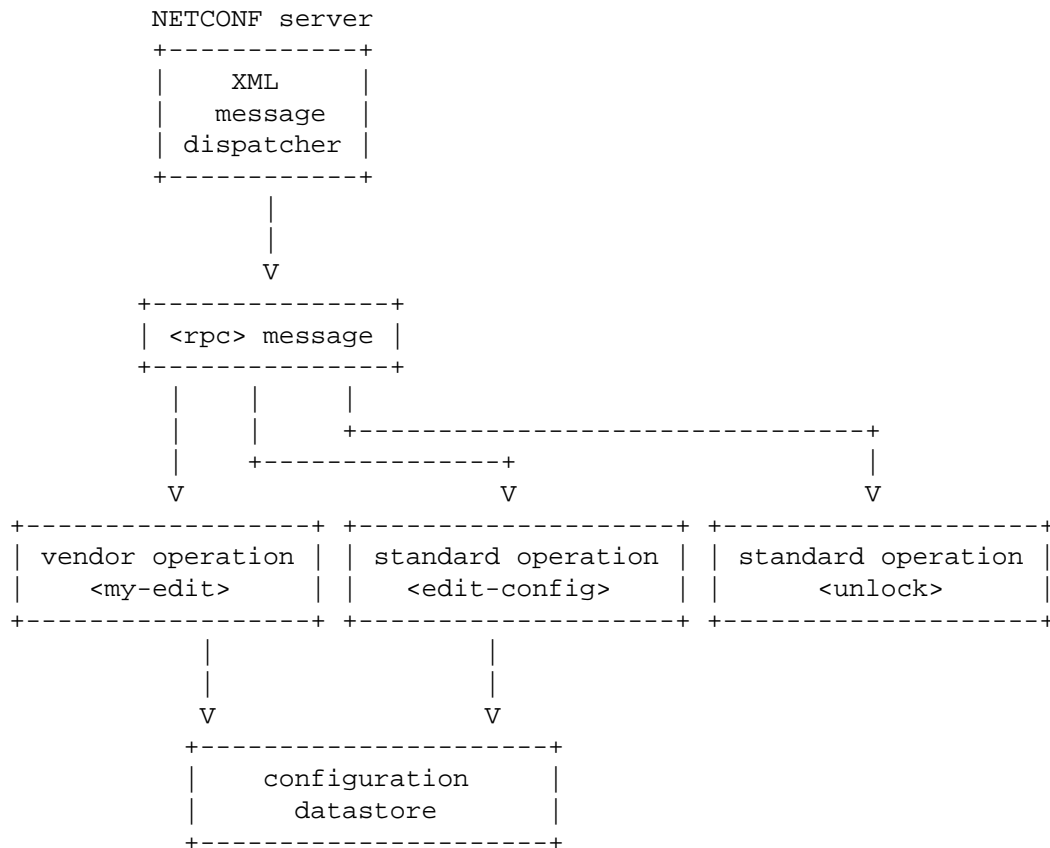
                           Figure 3

   Access control begins with the message dispatcher.

   After the server validates the <rpc> element and determines the
   namespace URI and the element name of the protocol operation being
   requested, the server verifies that the user is authorized to invoke
   the protocol operation.

The server MUST separately authorize every protocol operation by
following these steps:

1. If the "enable-nacm" leaf is set to "false", then the protocol
    operation is permitted.

2. If the requesting session is identified as a recovery session,
    then the protocol operation is permitted.

3. If the requested operation is the NETCONF <close-session>
    protocol operation, then the protocol operation is permitted.

4. Check all the "group" entries to see if any of them contain a
    "user-name" entry that equals the username for the session
    making the request. If the "enable-external-groups" leaf is
    "true", add to these groups the set of groups provided by the
    transport layer.

5. If no groups are found, continue with step 10.

6. Process all rule-list entries, in the order they appear in the
    configuration. If a rule-list's "group" leaf-list does not
    match any of the user's groups, proceed to the next rule-list
    entry.

7. For each rule-list entry found, process all rules, in order,
    until a rule that matches the requested access operation is
    found. A rule matches if all of the following criteria are met:

    * The rule's "module-name" leaf is "*" or equals the name of
       the YANG module where the protocol operation is defined.

    * Either (1) the rule does not have a "rule-type" defined or
       (2) the "rule-type" is "protocol-operation" and the
       "rpc-name" is "*" or equals the name of the requested
       protocol operation.

    * The rule's "access-operations" leaf has the "exec" bit set or
       has the special value "*".

8. If a matching rule is found, then the "action" leaf is checked.
    If it is equal to "permit", then the protocol operation is
    permitted; otherwise, it is denied.

9. At this point, no matching rule was found in any rule-list
    entry.

   10. If the requested protocol operation is defined in a YANG module
        advertised in the server capabilities and the "rpc" statement
        contains a "nacm:default-deny-all" statement, then the protocol
        operation is denied.

   11. If the requested protocol operation is the NETCONF
        <kill-session> or <delete-config>, then the protocol operation
        is denied.

   12. If the "exec-default" leaf is set to "permit", then permit the
        protocol operation; otherwise, deny the request.

   If the user is not authorized to invoke the protocol operation, then
   an <rpc-error> is generated with the following information:

   error-tag: access-denied

   error-path: Identifies the requested protocol operation. The
      following example represents the <edit-config> protocol operation
      in the NETCONF base namespace:

         <error-path
           xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
             /nc:rpc/nc:edit-config
         </error-path>

   If a datastore is accessed, either directly or as a side effect of
   the protocol operation, then the server MUST intercept the access
   operation and make sure that the user is authorized to perform the
   requested access operation on the specified data, as defined in
   Section 3.4.5.

3.4.5. Data Node Access Validation

   If (1) a data node within a datastore is accessed or (2) an action or
   notification is tied to a data node, then the server MUST ensure that
   the user is authorized to perform the requested "read", "create",
   "update", "delete", or "execute" access operation on the specified
   data node.

   If an action is requested to be executed, the server MUST ensure that
   the user is authorized to perform the "execute" access operation on
   the requested action.

   If a notification tied to a data node is generated, the server MUST
   ensure that the user is authorized to perform the "read" access
   operation on the requested notification.

The data node access request is authorized by following these steps:

1. If the "enable-nacm" leaf is set to "false", then the access
    operation is permitted.

2. If the requesting session is identified as a recovery session,
    then the access operation is permitted.

3. Check all the "group" entries to see if any of them contain a
    "user-name" entry that equals the username for the session
    making the request. If the "enable-external-groups" leaf is
    "true", add to these groups the set of groups provided by the
    transport layer.

4. If no groups are found, continue with step 9.

5. Process all rule-list entries, in the order they appear in the
    configuration. If a rule-list's "group" leaf-list does not
    match any of the user's groups, proceed to the next rule-list
    entry.

6. For each rule-list entry found, process all rules, in order,
    until a rule that matches the requested access operation is
    found. A rule matches if all of the following criteria are met:

    * The rule's "module-name" leaf is "*" or equals the name of
       the YANG module where the requested data node is defined.

    * Either (1) the rule does not have a "rule-type" defined or
       (2) the "rule-type" is "data-node" and the "path" matches the
       requested data node, action node, or notification node. A
       path is considered to match if the requested node is the node
       specified by the path or is a descendant node of the path.

    * For a "read" access operation, the rule's "access-operations"
       leaf has the "read" bit set or has the special value "*".

    * For a "create" access operation, the rule's
       "access-operations" leaf has the "create" bit set or has the
       special value "*".

    * For a "delete" access operation, the rule's
       "access-operations" leaf has the "delete" bit set or has the
       special value "*".

  * For an "update" access operation, the rule's
     "access-operations" leaf has the "update" bit set or has the
     special value "*".

  * For an "execute" access operation, the rule's
     "access-operations" leaf has the "exec" bit set or has the
     special value "*".

7. If a matching rule is found, then the "action" leaf is checked.
    If it is equal to "permit", then the data node access is
    permitted; otherwise, it is denied. For a "read" access
    operation, "denied" means that the requested data is not
    returned in the reply.

8. At this point, no matching rule was found in any rule-list
    entry.

9. For a "read" access operation, if the requested data node is
    defined in a YANG module advertised in the server capabilities
    and the data definition statement contains a
    "nacm:default-deny-all" statement, then the requested data node
    and all its descendants are not included in the reply.

10. For a "write" access operation, if the requested data node is
    defined in a YANG module advertised in the server capabilities
    and the data definition statement contains a
    "nacm:default-deny-write" or a "nacm:default-deny-all"
    statement, then the access request is denied for the data node
    and all its descendants.

11. For a "read" access operation, if the "read-default" leaf is set
    to "permit", then include the requested data node in the reply;
    otherwise, do not include the requested data node or any of its
    descendants in the reply.

12. For a "write" access operation, if the "write-default" leaf is
    set to "permit", then permit the data node access request;
    otherwise, deny the request.

13. For an "execute" access operation, if the "exec-default" leaf is
    set to "permit", then permit the request; otherwise, deny the
    request.

3.4.6. Outgoing <notification> Authorization

   Configuration of access control rules specifically for descendant
   nodes of the notification event type are outside the scope of this
   document. If the user is authorized to receive the notification
   event type, then it is also authorized to receive any data it
   contains.

   If the notification is specified within a data subtree, as specified
   in [RFC7950], then read access to the notification is required.
   Processing continues as described in Section 3.4.5.

   The following figure shows the conceptual message processing model
   for outgoing <notification> messages.

```
                        NETCONF server
                       +------------+
                       |    XML     |
                       |  message   |
                       | generator  |
                       +------------+
                             ^
                             |
                  +----------------+
                  | <notification> |
                  |   generator    |
                  +----------------+
                             ^
                             |
                  +================+
                  | <notification> |
                  |  access control|
                  |   <eventType>  |
                  +================+
                             ^
                             |
             +------------------------+
             | server instrumentation |
             +------------------------+
                        |    ^
                        V    |
              +---------------------+
              |    configuration    |
              |      datastore      |
              +---------------------+
```
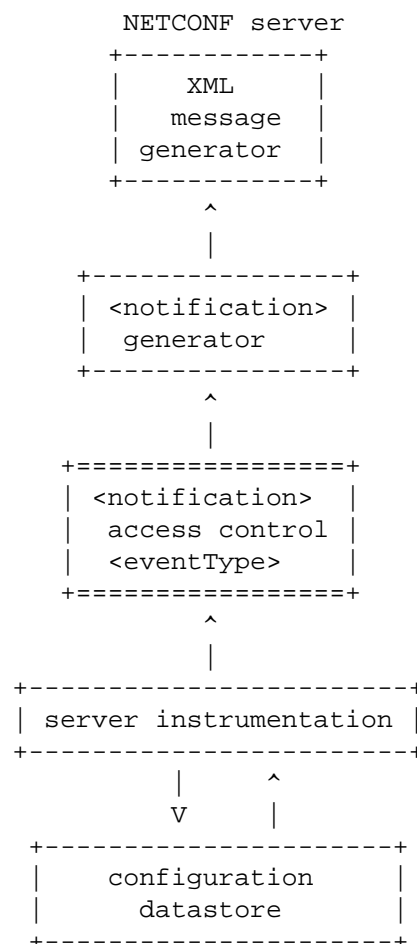
                           Figure 4

The generation of a notification for a specific subscription
[RFC5277] is authorized by following these steps:

1.   If the "enable-nacm" leaf is set to "false", then the
     notification is permitted.

2.   If the session is identified as a recovery session, then the
     notification is permitted.

3.   If the notification is the NETCONF <replayComplete> or
     <notificationComplete> event type [RFC5277], then the
     notification is permitted.

4.   Check all the "group" entries to see if any of them contain a
     "user-name" entry that equals the username for the session
     making the request. If the "enable-external-groups" leaf is
     "true", add to these groups the set of groups provided by the
     transport layer.

5.   If no groups are found, continue with step 10.

6.   Process all rule-list entries, in the order they appear in the
     configuration. If a rule-list's "group" leaf-list does not
     match any of the user's groups, proceed to the next rule-list
     entry.

7.   For each rule-list entry found, process all rules, in order,
     until a rule that matches the requested access operation is
     found. A rule matches if all of the following criteria are met:

     *   The rule's "module-name" leaf is "*" or equals the name of
         the YANG module where the notification is defined.

     *   Either (1) the rule does not have a "rule-type" defined or
         (2) the "rule-type" is "notification" and the
         "notification-name" is "*" or equals the name of the
         notification.

     *   The rule's "access-operations" leaf has the "read" bit set or
         has the special value "*".

8.   If a matching rule is found, then the "action" leaf is checked.
     If it is equal to "permit", then permit the notification;
     otherwise, drop the notification for the associated
     subscription.

9.   Otherwise, no matching rule was found in any rule-list entry.

   10. If the requested notification is defined in a YANG module
        advertised in the server capabilities and the "notification"
        statement contains a "nacm:default-deny-all" statement, then the
        notification is dropped for the associated subscription.

   11. If the "read-default" leaf is set to "permit", then permit the
        notification; otherwise, drop the notification for the
        associated subscription.

3.5. Data Model Definitions

3.5.1. Data Organization

   The following diagram highlights the contents and structure of the
   NACM YANG module.

   module: ietf-netconf-acm
     +--rw nacm
        +--rw enable-nacm?   boolean
        +--rw read-default?   action-type
        +--rw write-default?   action-type
        +--rw exec-default?   action-type
        +--rw enable-external-groups? boolean
        +--ro denied-operations   yang:zero-based-counter32
        +--ro denied-data-writes   yang:zero-based-counter32
        +--ro denied-notifications   yang:zero-based-counter32
        +--rw groups
        | +--rw group* [name]
        | +--rw name   group-name-type
        | +--rw user-name*   user-name-type
        +--rw rule-list* [name]
           +--rw name   string
           +--rw group*   union
           +--rw rule* [name]
              +--rw name   string
              +--rw module-name?   union
              +--rw (rule-type)?
              | +--:(protocol-operation)
              | | +--rw rpc-name?   union
              | +--:(notification)
              | | +--rw notification-name?   union
              | +--:(data-node)
              | +--rw path   node-instance-identifier
              +--rw access-operations?   union
              +--rw action   action-type
              +--rw comment?   string

3.5.2. YANG Module

   The following YANG module specifies the normative NETCONF content
   that MUST be supported by the server.

   The "ietf-netconf-acm" YANG module imports typedefs from [RFC6991].

   <CODE BEGINS> file "ietf-netconf-acm@2018-02-14.yang"
   module ietf-netconf-acm {

     namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-acm";

     prefix nacm;

     import ietf-yang-types {
       prefix yang;
     }

     organization
       "IETF NETCONF (Network Configuration) Working Group";

     contact
       "WG Web: <https://datatracker.ietf.org/wg/netconf/>
        WG List: <mailto:netconf@ietf.org>

        Author: Andy Bierman
                  <mailto:andy@yumaworks.com>

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

     description
       "Network Configuration Access Control Model.

        Copyright (c) 2012 - 2018 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
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC 8341; see
        the RFC itself for full legal notices.";

```
      revision "2018-02-14" {
        description
          "Added support for YANG 1.1 actions and notifications tied to
           data nodes.  Clarified how NACM extensions can be used by
           other data models.";
        reference
          "RFC 8341: Network Configuration Access Control Model";
      }

      revision "2012-02-22" {
        description
          "Initial version.";
        reference
          "RFC 6536: Network Configuration Protocol (NETCONF)
                     Access Control Model";
      }

      /*
       * Extension statements
       */

      extension default-deny-write {
        description
          "Used to indicate that the data model node
           represents a sensitive security system parameter.

           If present, the NETCONF server will only allow the designated
           'recovery session' to have write access to the node.  An
           explicit access control rule is required for all other users.

           If the NACM module is used, then it must be enabled (i.e.,
           /nacm/enable-nacm object equals 'true'), or this extension
           is ignored.

           The 'default-deny-write' extension MAY appear within a data
           definition statement.  It is ignored otherwise.";
      }

      extension default-deny-all {
        description
          "Used to indicate that the data model node
           controls a very sensitive security system parameter.

           If present, the NETCONF server will only allow the designated
           'recovery session' to have read, write, or execute access to
           the node.  An explicit access control rule is required for all
           other users.
```

```
        If the NACM module is used, then it must be enabled (i.e.,
        /nacm/enable-nacm object equals 'true'), or this extension
        is ignored.

        The 'default-deny-all' extension MAY appear within a data
        definition statement, 'rpc' statement, or 'notification'
        statement.  It is ignored otherwise.";
  }

  /*
   * Derived types
   */

  typedef user-name-type {
    type string {
      length "1..max";
    }
    description
      "General-purpose username string.";
  }

  typedef matchall-string-type {
    type string {
      pattern '\*';
    }
    description
      "The string containing a single asterisk '*' is used
       to conceptually represent all possible values
       for the particular leaf using this data type.";
  }

  typedef access-operations-type {
    type bits {
      bit create {
        description
          "Any protocol operation that creates a
           new data node.";
      }
      bit read {
        description
          "Any protocol operation or notification that
           returns the value of a data node.";
      }
      bit update {
        description
          "Any protocol operation that alters an existing
           data node.";
      }
```

```
        bit delete {
          description
            "Any protocol operation that removes a data node.";
        }
        bit exec {
          description
            "Execution access to the specified protocol operation.";
        }
      }
      description
        "Access operation.";
    }

    typedef group-name-type {
      type string {
        length "1..max";
        pattern '[^\*].*';
      }
      description
        "Name of administrative group to which
         users can be assigned.";
    }

    typedef action-type {
      type enumeration {
        enum permit {
          description
            "Requested action is permitted.";
        }
        enum deny {
          description
            "Requested action is denied.";
        }
      }
      description
        "Action taken by the server when a particular
         rule matches.";
    }

    typedef node-instance-identifier {
      type yang:xpath1.0;
      description
        "Path expression used to represent a special
         data node, action, or notification instance-identifier
         string.

         A node-instance-identifier value is an
         unrestricted YANG instance-identifier expression.
```

```
            All the same rules as an instance-identifier apply,
            except that predicates for keys are optional.  If a key
            predicate is missing, then the node-instance-identifier
            represents all possible server instances for that key.

            This XML Path Language (XPath) expression is evaluated in the
            following context:

               o  The set of namespace declarations are those in scope on
                  the leaf element where this type is used.

               o  The set of variable bindings contains one variable,
                  'USER', which contains the name of the user of the
                  current session.

               o  The function library is the core function library, but
                  note that due to the syntax restrictions of an
                  instance-identifier, no functions are allowed.

               o  The context node is the root node in the data tree.

            The accessible tree includes actions and notifications tied
            to data nodes.";
      }

      /*
       * Data definition statements
       */

      container nacm {
        nacm:default-deny-all;

        description
          "Parameters for NETCONF access control model.";

        leaf enable-nacm {
          type boolean;
          default "true";
          description
            "Enables or disables all NETCONF access control
             enforcement.  If 'true', then enforcement
             is enabled.  If 'false', then enforcement
             is disabled.";
        }
```

```
    leaf read-default {
      type action-type;
      default "permit";
      description
        "Controls whether read access is granted if
         no appropriate rule is found for a
         particular read request.";
    }

    leaf write-default {
      type action-type;
      default "deny";
      description
        "Controls whether create, update, or delete access
         is granted if no appropriate rule is found for a
         particular write request.";
    }

    leaf exec-default {
      type action-type;
      default "permit";
      description
        "Controls whether exec access is granted if no appropriate
         rule is found for a particular protocol operation request.";
    }

    leaf enable-external-groups {
      type boolean;
      default "true";
      description
        "Controls whether the server uses the groups reported by the
         NETCONF transport layer when it assigns the user to a set of
         NACM groups.  If this leaf has the value 'false', any group
         names reported by the transport layer are ignored by the
         server.";
    }

    leaf denied-operations {
      type yang:zero-based-counter32;
      config false;
      mandatory true;
      description
        "Number of times since the server last restarted that a
         protocol operation request was denied.";
    }
```

```
      leaf denied-data-writes {
        type yang:zero-based-counter32;
        config false;
        mandatory true;
        description
          "Number of times since the server last restarted that a
           protocol operation request to alter
           a configuration datastore was denied.";
      }

      leaf denied-notifications {
        type yang:zero-based-counter32;
        config false;
        mandatory true;
        description
          "Number of times since the server last restarted that
           a notification was dropped for a subscription because
           access to the event type was denied.";
      }

      container groups {
        description
          "NETCONF access control groups.";

        list group {
          key name;

          description
            "One NACM group entry.  This list will only contain
             configured entries, not any entries learned from
             any transport protocols.";

          leaf name {
            type group-name-type;
            description
              "Group name associated with this entry.";
          }

          leaf-list user-name {
            type user-name-type;
            description
              "Each entry identifies the username of
               a member of the group associated with
               this entry.";
          }
        }
      }
```

```
        list rule-list {
          key name;
          ordered-by user;
          description
            "An ordered collection of access control rules.";

          leaf name {
            type string {
              length "1..max";
            }
            description
              "Arbitrary name assigned to the rule-list.";
          }
          leaf-list group {
            type union {
              type matchall-string-type;
              type group-name-type;
            }
            description
              "List of administrative groups that will be
               assigned the associated access rights
               defined by the 'rule' list.

               The string '*' indicates that all groups apply to the
               entry.";
          }

          list rule {
            key name;
            ordered-by user;
            description
              "One access control rule.

               Rules are processed in user-defined order until a match is
               found.  A rule matches if 'module-name', 'rule-type', and
               'access-operations' match the request.  If a rule
               matches, the 'action' leaf determines whether or not
               access is granted.";

            leaf name {
              type string {
                length "1..max";
              }
              description
                "Arbitrary name assigned to the rule.";
            }
```

```
            leaf module-name {
              type union {
                type matchall-string-type;
                type string;
              }
              default "*";
              description
                "Name of the module associated with this rule.

                 This leaf matches if it has the value '*' or if the
                 object being accessed is defined in the module with the
                 specified module name.";
            }
            choice rule-type {
              description
                "This choice matches if all leafs present in the rule
                 match the request.  If no leafs are present, the
                 choice matches all requests.";
              case protocol-operation {
                leaf rpc-name {
                  type union {
                    type matchall-string-type;
                    type string;
                  }
                   description
                    "This leaf matches if it has the value '*' or if
                     its value equals the requested protocol operation
                     name.";
                }
              }
              case notification {
                leaf notification-name {
                  type union {
                    type matchall-string-type;
                    type string;
                  }
                   description
                    "This leaf matches if it has the value '*' or if its
                     value equals the requested notification name.";
                }
              }
```

```
          case data-node {
            leaf path {
              type node-instance-identifier;
              mandatory true;
              description
                "Data node instance-identifier associated with the
                 data node, action, or notification controlled by
                 this rule.

                 Configuration data or state data
                 instance-identifiers start with a top-level
                 data node.  A complete instance-identifier is
                 required for this type of path value.

                 The special value '/' refers to all possible
                 datastore contents.";
            }
          }
        }

        leaf access-operations {
          type union {
            type matchall-string-type;
            type access-operations-type;
          }
          default "*";
          description
            "Access operations associated with this rule.

             This leaf matches if it has the value '*' or if the
             bit corresponding to the requested operation is set.";
        }

        leaf action {
          type action-type;
          mandatory true;
          description
            "The access control action associated with the
             rule.  If a rule has been determined to match a
             particular request, then this object is used
             to determine whether to permit or deny the
             request.";
        }
```

```
          leaf comment {
            type string;
            description
              "A textual description of the access rule.";
          }
        }
      }
    }
  }

  <CODE ENDS>
```

4. IANA Considerations

   This document reuses the URI for "ietf-netconf-acm" in the "IETF XML
   Registry".

   This document updates the module registration in the "YANG Module
   Names" registry to reference this RFC instead of RFC 6536 for
   "ietf-netconf-acm". Following the format in [RFC6020], the following
   has been registered.

        Name: ietf-netconf-acm
        Namespace: urn:ietf:params:xml:ns:yang:ietf-netconf-acm
        Prefix: nacm
        Reference: RFC 8341

5. Security Considerations

   The YANG module specified in this document defines a schema for data
   that is designed to be accessed via network management protocols such
   as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer
   is the secure transport layer, and the mandatory-to-implement secure
   transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer
   is HTTPS, and the mandatory-to-implement secure transport is TLS
   [RFC5246].

   The NETCONF access control model [RFC8341] provides the means to
   restrict access for particular NETCONF or RESTCONF users to a
   preconfigured subset of all available NETCONF or RESTCONF protocol
   operations and content.

   There is a risk related to the lack of access control enforcement for
   the RESTCONF OPTIONS and PATCH methods. The risk here is that the
   response to OPTIONS and PATCH may vary based on the presence or
   absence of a resource corresponding to the URL's path. If this is
   the case, then it can be used to trivially probe for the presence or
   absence of values within a tree. Therefore, a server MUST NOT vary

   its responses based on the existence of the underlying resource,
   which would indicate the presence or absence of resource instances.
   In particular, servers should not expose any instance information
   before ensuring that the client has the necessary access permissions
   to obtain that information. In such cases, servers are expected to
   always return the "access-denied" error response.

   There are a number of data nodes defined in this YANG module that are
   writable/creatable/deletable (i.e., config true, which is the
   default). These data nodes may be considered sensitive or vulnerable
   in some network environments. Write operations (e.g., edit-config)
   to these data nodes without proper protection can have a negative
   effect on network operations. These are the subtrees and data nodes
   and their sensitivity/vulnerability:

   o /nacm: The entire /nacm subtree is related to security. Refer to
      the following sections for more details.

   This section highlights the issues for an administrator to consider
   when configuring a NETCONF server with the NACM.

5.1. NACM Configuration and Monitoring Considerations

   Configuration of the access control system is highly sensitive to
   system security. A server may choose not to allow any user
   configuration to some portions of it, such as the global security
   level or the groups that allowed access to system resources.

   By default, NACM enforcement is enabled. By default, "read" access
   to all datastore contents is enabled (unless "nacm:default-deny-all"
   is specified for the data definition), and "exec" access is enabled
   for safe protocol operations. An administrator needs to ensure that
   the NACM is enabled and also decide if the default access parameters
   are set appropriately. Make sure that the following data nodes are
   properly configured:

   o /nacm/enable-nacm (default "true")

   o /nacm/read-default (default "permit")

   o /nacm/write-default (default "deny")

   o /nacm/exec-default (default "permit")

   An administrator needs to restrict write access to all configurable
   objects within this data model.

If write access is allowed for configuration of access control rules,
then care needs to be taken not to disrupt the access control
enforcement. For example, if the NACM access control rules are
edited directly within the running configuration datastore (i.e.,
:writable-running capability is supported and used), then care needs
to be taken not to allow unintended access while the edits are being
done.

An administrator needs to make sure that the translation from a
transport- or implementation-dependent user identity to a NACM
username is unique and correct. This requirement is specified in
detail in Section 2.2 of [RFC6241].

An administrator needs to be aware that the YANG data structures
representing access control rules (/nacm/rule-list and
/nacm/rule-list/rule) are ordered by the client. The server will
evaluate the access control rules according to their relative
conceptual order within the running configuration datastore.

Note that the /nacm/groups data structure contains the administrative
group names used by the server. These group names may be configured
locally and/or provided through an external protocol, such as RADIUS
[RFC2865] [RFC5607].

An administrator needs to be aware of the security properties of any
external protocol used by the transport layer to determine group
names. For example, if this protocol does not protect against
man-in-the-middle attacks, an attacker might be able to inject group
names that are configured in the NACM so that a user gets more
permissions than it should. In such cases, the administrator may
wish to disable the usage of such group names by setting
/nacm/enable-external-groups to "false".

Some of the readable data nodes in this YANG module may be considered
sensitive or vulnerable in some network environments. It is thus
important to control read access (e.g., via get, get-config, or
notification) to these data nodes. These are the subtrees and data
nodes and their sensitivity/vulnerability:

o   /nacm/enable-nacm

o   /nacm/read-default

o   /nacm/write-default

o   /nacm/exec-default

o   /nacm/enable-external-groups

   o /nacm/groups

   o /nacm/rule-list

   An administrator needs to restrict read access to the above-listed
   objects within this data model, as they reveal access control
   configuration that could be considered sensitive.

5.2. General Configuration Issues

   There is a risk that invocation of non-standard protocol operations
   will have undocumented side effects. An administrator needs to
   construct access control rules such that the configuration datastore
   is protected from such side effects.

   It is possible for a session with some write access (e.g., allowed to
   invoke <edit-config>), but without any access to a particular
   datastore subtree containing sensitive data, to determine the
   presence or non-presence of that data. This can be done by
   repeatedly issuing some sort of edit request (create, update, or
   delete) and possibly receiving "access-denied" errors in response.
   These "fishing" attacks can identify the presence or non-presence of
   specific sensitive data even without the "error-path" field being
   present within the <rpc-error> response.

   It may be possible for the set of NETCONF capabilities on the server
   to change over time. If so, then there is a risk that new protocol
   operations, notifications, and/or datastore content have been added
   to the device. An administrator needs to be sure that the access
   control rules are correct for the new content in this case.
   Mechanisms to detect NETCONF capability changes on a specific device
   are outside the scope of this document.

   It is possible that the data model definition itself (e.g., a YANG
   when-stmt) will help an unauthorized session determine the presence
   or even value of sensitive data nodes by examining the presence and
   values of different data nodes.

   It is possible that the data model definition itself (e.g., a YANG
   when-stmt or choice-stmt) will allow a session to implicitly create
   or delete nodes that the session does not have write access to as an
   implicit side effect from the processing of an allowed <edit-config>
   operation.

There is a risk that non-standard protocol operations, or even the
standard <get> protocol operation, may return data that "aliases" or
"copies" sensitive data from a different data object. There may
simply be multiple data model definitions that expose or even
configure the same underlying system instrumentation.

A data model may contain external keys (e.g., YANG leafref), which
expose values from a different data structure. An administrator
needs to be aware of sensitive data models that contain leafref
nodes. This entails finding all the leafref objects that "point" at
the sensitive data (i.e., "path-stmt" values) that implicitly or
explicitly includes the sensitive data node.

It is beyond the scope of this document to define access control
enforcement procedures for underlying device instrumentation that may
exist to support the NETCONF server operation. An administrator can
identify each protocol operation that the server provides and decide
if it needs any access control applied to it.

This document incorporates the optional use of a recovery session
mechanism, which can be used to bypass access control enforcement in
emergencies such as NACM configuration errors that disable all access
to the server. The configuration and identification of such a
recovery session mechanism are implementation specific and are
outside the scope of this document. An administrator needs to be
aware of any recovery session mechanisms available on the device and
make sure that they are used appropriately.

It is possible for a session to disrupt configuration management,
even without any write access to the configuration, by locking the
datastore. This may be done to ensure that all or part of the
configuration remains stable while it is being retrieved, or it may
be done as a "denial-of-service" attack. There is no way for the
server to know the difference. An administrator may wish to restrict
"exec" access to the following protocol operations:

o <lock>

o <unlock>

o <partial-lock>

o <partial-unlock>

5.3. Data Model Design Considerations

   Designers need to clearly identify any sensitive data, notifications,
   or protocol operations defined within a YANG module. For such
   definitions, a "nacm:default-deny-write" or "nacm:default-deny-all"
   statement ought to be present, in addition to a clear description of
   the security risks.

   Protocol operations need to be properly documented by the data model
   designer so that it is clear to administrators what data nodes (if
   any) are affected by the protocol operation and what information (if
   any) is returned in the <rpc-reply> message.

   Data models ought to be designed so that different access levels for
   input parameters to protocol operations are not required. The use of
   generic protocol operations should be avoided, and if different
   access levels are needed, separate protocol operations should be
   defined instead.

6. References

6.1. 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>.

   [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.2", RFC 5246,
              DOI 10.17487/RFC5246, August 2008,
              <https://www.rfc-editor.org/info/rfc5246>.

   [RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event
              Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008,
              <https://www.rfc-editor.org/info/rfc5277>.

   [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for
              the Network Configuration Protocol (NETCONF)", RFC 6020,
              DOI 10.17487/RFC6020, October 2010,
              <https://www.rfc-editor.org/info/rfc6020>.

   [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
              and A. Bierman, Ed., "Network Configuration Protocol
              (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,
              <https://www.rfc-editor.org/info/rfc6241>.

   [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure
              Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011,
              <https://www.rfc-editor.org/info/rfc6242>.

   [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types",
              RFC 6991, DOI 10.17487/RFC6991, July 2013,
              <https://www.rfc-editor.org/info/rfc6991>.

   [RFC7230] Fielding, R., Ed., and J. Reschke, Ed., "Hypertext
              Transfer Protocol (HTTP/1.1): Message Syntax and Routing",
              RFC 7230, DOI 10.17487/RFC7230, June 2014,
              <https://www.rfc-editor.org/info/rfc7230>.

   [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>.

   [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF
              Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017,
              <https://www.rfc-editor.org/info/rfc8040>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in
              RFC 2119 Key Words", BCP 14, RFC 8174,
              DOI 10.17487/RFC8174, May 2017,
              <https://www.rfc-editor.org/info/rfc8174>.

   [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>.

   [W3C.REC-xml-20081126]
              Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., and
              F. Yergeau, "Extensible Markup Language (XML) 1.0
              (Fifth Edition)", World Wide Web Consortium Recommendation
              REC-xml-20081126, November 2008,
              <https://www.w3.org/TR/2008/REC-xml-20081126>.

6.2. Informative References

   [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson,
              "Remote Authentication Dial In User Service (RADIUS)",
              RFC 2865, DOI 10.17487/RFC2865, June 2000,
              <https://www.rfc-editor.org/info/rfc2865>.

   [RFC5607] Nelson, D. and G. Weber, "Remote Authentication Dial-In
              User Service (RADIUS) Authorization for Network Access
              Server (NAS) Management", RFC 5607, DOI 10.17487/RFC5607,
              July 2009, <https://www.rfc-editor.org/info/rfc5607>.

   [YANG-SEC] IETF, "YANG Security Guidelines", <https://trac.ietf.org/
              trac/ops/wiki/yang-security-guidelines>.

Appendix A. Usage Examples

   The following XML [W3C.REC-xml-20081126] snippets are provided as
   examples only, to demonstrate how the NACM can be configured to
   perform some access control tasks.

A.1. <groups> Example

   There needs to be at least one <group> entry in order for any of the
   access control rules to be useful.

   The following XML shows arbitrary groups and is not intended to
   represent any particular use case.

   <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
     <groups>
       <group>
         <name>admin</name>
         <user-name>admin</user-name>
         <user-name>andy</user-name>
       </group>

       <group>
         <name>limited</name>
         <user-name>wilma</user-name>
         <user-name>bam-bam</user-name>
       </group>

       <group>
         <name>guest</name>
         <user-name>guest</user-name>
         <user-name>guest@example.com</user-name>
       </group>
     </groups>
   </nacm>

   This example shows three groups:

   admin: The "admin" group contains two users named "admin" and
      "andy".

   limited: The "limited" group contains two users named "wilma" and
      "bam-bam".

   guest: The "guest" group contains two users named "guest" and
      "guest@example.com".

A.2. Module Rule Example

   Module rules are used to control access to all the content defined in
   a specific module. A module rule has the "module-name" leaf set but
   no nodes from the "rule-type" choice set.

```
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <rule-list>
    <name>guest-acl</name>
    <group>guest</group>

    <rule>
      <name>deny-ncm</name>
      <module-name>ietf-netconf-monitoring</module-name>
      <access-operations>*</access-operations>
      <action>deny</action>
      <comment>
          Do not allow guests any access to the NETCONF
          monitoring information.
      </comment>
    </rule>
  </rule-list>

  <rule-list>
    <name>limited-acl</name>
    <group>limited</group>

    <rule>
      <name>permit-ncm</name>
      <module-name>ietf-netconf-monitoring</module-name>
      <access-operations>read</access-operations>
      <action>permit</action>
      <comment>
          Allow read access to the NETCONF
          monitoring information.
      </comment>
    </rule>
    <rule>
      <name>permit-exec</name>
      <module-name>*</module-name>
      <access-operations>exec</access-operations>
      <action>permit</action>
      <comment>
          Allow invocation of the
          supported server operations.
      </comment>
    </rule>
  </rule-list>
```

```
   <rule-list>
     <name>admin-acl</name>
     <group>admin</group>

     <rule>
       <name>permit-all</name>
       <module-name>*</module-name>
       <access-operations>*</access-operations>
       <action>permit</action>
       <comment>
           Allow the 'admin' group complete access to all
           operations and data.
       </comment>
     </rule>
   </rule-list>
</nacm>
```

   This example shows four module rules:

   deny-ncm: This rule prevents the "guest" group from reading any
      monitoring information in the "ietf-netconf-monitoring" YANG
      module.

   permit-ncm: This rule allows the "limited" group to read the
      "ietf-netconf-monitoring" YANG module.

   permit-exec: This rule allows the "limited" group to invoke any
      protocol operation supported by the server.

   permit-all: This rule allows the "admin" group complete access to
      all content in the server. No subsequent rule will match for the
      "admin" group because of this module rule.

A.3. Protocol Operation Rule Example

   Protocol operation rules are used to control access to a specific
   protocol operation.

```
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <rule-list>
    <name>guest-limited-acl</name>
    <group>limited</group>
    <group>guest</group>

    <rule>
      <name>deny-kill-session</name>
      <module-name>ietf-netconf</module-name>
      <rpc-name>kill-session</rpc-name>
      <access-operations>exec</access-operations>
      <action>deny</action>
      <comment>
        Do not allow the 'limited' group or the 'guest' group
        to kill another session.
      </comment>
    </rule>
    <rule>
      <name>deny-delete-config</name>
      <module-name>ietf-netconf</module-name>
      <rpc-name>delete-config</rpc-name>
      <access-operations>exec</access-operations>
      <action>deny</action>
      <comment>
        Do not allow the 'limited' group or the 'guest' group
        to delete any configurations.
      </comment>
    </rule>
  </rule-list>
```

```
   <rule-list>
     <name>limited-acl</name>
     <group>limited</group>

     <rule>
       <name>permit-edit-config</name>
       <module-name>ietf-netconf</module-name>
       <rpc-name>edit-config</rpc-name>
       <access-operations>exec</access-operations>
       <action>permit</action>
       <comment>
         Allow the 'limited' group to edit the configuration.
       </comment>
     </rule>
   </rule-list>
</nacm>
```

   This example shows three protocol operation rules:

   deny-kill-session: This rule prevents the "limited" group or the
      "guest" group from invoking the NETCONF <kill-session> protocol
      operation.

   deny-delete-config: This rule prevents the "limited" group or the
      "guest" group from invoking the NETCONF <delete-config> protocol
      operation.

   permit-edit-config: This rule allows the "limited" group to invoke
      the NETCONF <edit-config> protocol operation. This rule will have
      no real effect unless the "exec-default" leaf is set to "deny".

A.4. Data Node Rule Example

   Data node rules are used to control access to specific (config and
   non-config) data nodes within the NETCONF content provided by the
   server.

```
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <rule-list>
    <name>guest-acl</name>
    <group>guest</group>

    <rule>
      <name>deny-nacm</name>
      <path xmlns:n="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
        /n:nacm
      </path>
      <access-operations>*</access-operations>
      <action>deny</action>
      <comment>
        Deny the 'guest' group any access to the /nacm data.
      </comment>
    </rule>
  </rule-list>

  <rule-list>
    <name>limited-acl</name>
    <group>limited</group>

    <rule>
      <name>permit-acme-config</name>
      <path xmlns:acme="http://example.com/ns/netconf">
        /acme:acme-netconf/acme:config-parameters
      </path>
      <access-operations>
        read create update delete
      </access-operations>
      <action>permit</action>
      <comment>
        Allow the 'limited' group complete access to the acme
        NETCONF configuration parameters.  Showing long form
        of 'access-operations' instead of shorthand.
      </comment>
    </rule>
  </rule-list>
```

```
       <rule-list>
         <name>guest-limited-acl</name>
         <group>guest</group>
         <group>limited</group>

         <rule>
           <name>permit-dummy-interface</name>
           <path xmlns:acme="http://example.com/ns/itf">
             /acme:interfaces/acme:interface[acme:name='dummy']
           </path>
           <access-operations>read update</access-operations>
           <action>permit</action>
           <comment>
             Allow the 'limited' and 'guest' groups read
             and update access to the dummy interface.
           </comment>
         </rule>
       </rule-list>

       <rule-list>
         <name>admin-acl</name>
         <group>admin</group>
         <rule>
           <name>permit-interface</name>
           <path xmlns:acme="http://example.com/ns/itf">
             /acme:interfaces/acme:interface
           </path>
           <access-operations>*</access-operations>
           <action>permit</action>
           <comment>
             Allow the 'admin' group full access to all acme interfaces.
           </comment>
         </rule>
       </rule-list>
     </nacm>
```

   This example shows four data node rules:

   deny-nacm: This rule denies the "guest" group any access to the
      /nacm subtree.

   permit-acme-config: This rule gives the "limited" group read-write
      access to the acme <config-parameters>.

   permit-dummy-interface: This rule gives the "limited" and "guest"
      groups read-update access to the acme <interface> entry named
      "dummy". This entry cannot be created or deleted by these groups;
      it can only be altered.

   permit-interface: This rule gives the "admin" group read-write
      access to all acme <interface> entries.

A.5. Notification Rule Example

   Notification rules are used to control access to a specific
   notification event type.

```
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <rule-list>
    <name>sys-acl</name>
    <group>limited</group>
    <group>guest</group>

    <rule>
      <name>deny-config-change</name>
      <module-name>acme-system</module-name>
      <notification-name>sys-config-change</notification-name>
      <access-operations>read</access-operations>
      <action>deny</action>
      <comment>
        Do not allow the 'guest' group or the 'limited' group
        to receive config change events.
      </comment>
    </rule>
  </rule-list>
</nacm>
```

   This example shows one notification rule:

   deny-config-change: This rule prevents the "limited" group or the
      "guest" group from receiving the acme <sys-config-change>
      event type.

Authors' Addresses

   Andy Bierman
   YumaWorks
   685 Cochran St.
   Suite #160
   Simi Valley, CA 93065
   United States of America

   Email: andy@yumaworks.com

   Martin Bjorklund
   Tail-f Systems

   Email: mbj@tail-f.com