NETCONF
Internet-Draft
Intended status: Standards Track
Expires: July 10, 2008

B. Lengyel
Ericsson Hungary
M. Bjorklund
Tail-f Systems
January 07, 2008

Partial Lock RPC for NETCONF
draft-ietf-netconf-partial-lock-00

Status of this Memo

Copyright Notice

Abstract

The NETCONF protocol defines the lock and unlock RPCs that lock entire configuration datastores. In some situations, a way to lock only parts of a configuration datastore is required. This document defines a capability-based extension to the NETCONF protocol for locking portions of a configuration datastore.

Table of Contents

1. Introduction

The NETCONF protocol [RFC4741] describes the lock and unlock RPCs that operate on entire configuration datastores. Often, multiple management sessions need to be able to modify the configuration of a managed device in parallel. In these cases, locking only parts of a configuration datastore is needed. This document defines an extension to the NETCONF protocol to allow this.

The mechanism for partial locking will be based on the existing XPath filtering mechanisms.

Partial locking will be introduced as a capability to NETCONF.

1.1. Definition of Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Partial Locking Capability

2.1. Overview

The :partial-lock capability indicates that the device supports the locking of its configuration with a scope smaller then a complete configuration datastore. Partial locking covers configuration data, but not state data.

The system will ensure that configuration resources covered by the lock will not be modified by other NETCONF or non-NETCONF management operations such as SNMP and the CLI.

The duration of the partial lock is defined as beginning when the partial lock is granted and lasting until either the corresponding

<partial-unlock> operation succeeds or the NETCONF session terminates.

A NETCONF session MAY have multiple parts of one or more datastores locked using partial lock operations. The <partial-lock> operation returns a lock-id to identify each successfully acquired lock.

2.2. Dependencies

If the :xpath capability is supported, the filter expressions can be full XPath 1.0 expressions.

2.3. Capability Identifier

urn:ietf:params:netconf:capability:partial-lock:1.0

2.4. New Operations

2.4.1. <partial-lock>

The <partial-lock> operation allows the client to lock a portion of a data store. The portion to lock is specified by using XPath filter parameters in the <partial-lock> operation. Each XPath expression MUST return a node set.

The XPath filter expressions are evaluated only once at lock time, thereafter the scope of the lock is maintained as a set of nodes. If the configuration data is later altered in a way that would make the original XPath filter expressions evaluate to a different set of nodes, this does not affect the scope of the partial lock.

XPath is only used for the creation of the partial lock. Conceptually the scope of the lock is defined by the returned nodeset and not by the XPath expression.

A <partial-lock> operation MUST be handled atomically by the NETCONF server. The server either locks all requested parts of the data store or none.

If a node is locked by a session, only that same session will be able to modify that node or any node in the subtree underneath it.

If a top level node of a locked subtree is deleted, any other session can recreate it, as it is not covered by the lock anymore.

A partial lock MUST fail if:

- Any NETCONF session (including the current session) owns the global lock on the datastore.

- Any part of the scope to be locked is already locked by another management session/protocol, including other NETCONF sessions using the <partial-lock> or any other non-NETCONF management method.

- The NETCONF server implements access control and the locking user does not have at least some basic access rights, e.g., read rights, to all of the datastore section to be locked. The exact handling of access rights is outside the scope of this document, but it is assumed that there is an access control system that MAY

deny or allow the partial lock operation.

As with most locking systems, there is a possibility that two users trying to lock different parts of the configuration become dead-locked. To avoid this situation, clients SHOULD lock everything they need in one operation. If that operation still fails, the client SHOULD back down, release any already acquired locks, and retry the procedure after some time interval. The length of the interval should preferably be random to avoid repeated dead-locks when both (or all) clients back down and then repeat locking.

It is the intention to keep partial-locking simple, so when a partial lock is executed you get what you asked for: a set of nodes that are locked for writing. There are some other issues that are intentionally not addressed to for the sake of simplicity.

- Locking does not effect read operations.

- If a part of a datastore is locked, this has no effect on any unlocked parts of the datastore. If this is a problem e.g. the operators changes depend on data values in the unlocked part of the datastore, the operator should include these values in the

scope of the lock.

- An operator is allowed to edit the configuration both inside and outside the scope of a lock. It is the operator's responsibility to lock all parts of the datastore that are crucial for a specific management action.

Note: The <partial-lock> operation does not modify the global <lock> operation defined in the base NETCONF Protocol [RFC4741]. If part of a datastore is already locked by <partial-lock>, then a global lock for that datastore will fail even if the global lock is attempted by the same NETCONF session which owns the partial-lock.

Parameters:

target: Name of the configuration datastore of which a part will be locked. URIs are not accepted.

select: One or more 'select' elements each contain an XPath filter expression. The XPath expression is evaluated in a context where the context node is the root of the server's conceptual data model, and the set of namespace declarations are those in scope on the select element.

The filter MUST return a node set.

If the device does not support the :xpath capability, the XPath expression MUST be limited to an Instance Identifier expression [Editor's Note: add text or reference. An Instance Identifier is an absolute path expression in abbreviated syntax, where predicates are used only to specify values for nodes defined as keys to distinguish multiple instances.]

Example: Lock virtual router 1 and interface eth1

```
<nc:rpc
  xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
  xmlns="urn:ietf:params:xml:ns:netconf:partial-lock:1.0"
  xmlns:rte="http://example.com/ns/route">
  xmlns:if="http://example.com/ns/interface">
```

```
  nc:message-id="135">
    <partial-lock>
        <nc:running/>
        <select>/routing/virtualRouter['routerName=router1']</select>
        <select>/interfaces/['interfaceId=eth1']</select>
     </partial-lock>
</nc:rpc>

<nc:rpc-reply
  xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
  xmlns="urn:ietf:params:xml:ns:netconf:partial-lock:1.0"
  xmlns:rte="http://example.com/ns/route">
  xmlns:if="http://example.com/ns/interface">
  nc:message-id="135">
    <nc:data>
        <lock-id>127</lock-id>
    </nc:data>
</nc:rpc-reply>
```

Positive Response:

If the device was able to satisfy the request, an <rpc-reply> is sent with a <lock-id> element (lock identifier) in the <data> element.

Negative Response:

If a lock is already held on any node within the subtrees to be locked, the <error-tag> element will be 'lock-denied' and the <error-info> element will include the <session-id> of the lock owner. If the lock is held by a non-NETCONF entity, a <session-id> of 0 (zero) is included.

If the select filters return an empty node set, the <error-tag> will be 'operation-failed', and the <error-app-tag> will be 'no-matches'.

If any select filter returns anything but a node set, the <error-tag> will be 'invalid-value'.

If the :xpath capability is not supported and the XPath expression is not an Instance Identifier, the <error-tag> will be 'invalid-value'.

If access control denies the partial lock, the <error-tag> will be 'access-denied'.

2.4.2. <partial-unlock>

The operation unlocks a part of a datastore that was previously locked using <partial-lock> during the same session.

Parameters:

lock-id: Lock identifier to unlock; taken from a reply to a previous <partial-lock> operation.

Example: Unlock

```
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns="urn:ietf:params:xml:ns:netconf:partial-lock:1.0"
    nc:message-id="136">
    <partial-unlock>
        <lock-id>127</lock-id>
    </partial-unlock>
</nc:rpc>
```

Positive Response:

If the device was able to satisfy the request, an <rpc-reply> is sent that contains an <ok> element. A positive response MUST be sent even if all of the locked part of the datastore has already been deleted.

Negative Response:

If the <lock-id> parameter does not identify a lock which is owned by the session, an 'invalid-value' error is returned.

2.5. Modifications to Existing Operations

None.

2.6. Interactions with Other Capabilities

2.6.1. Writable-Running Capability

Partial locking of the running datastore can only be done if the :writable-running capability is supported by the device.

2.6.2. Candidate Configuration Capability

Partial locking of the candidate datastore can only be done if the :candidate capability is supported by the device. The partial locking of the candidate datastore does not depend on whether the datastore was modified or not.

2.6.3. Distinct Startup Capability

Partial locking of the startup datastore can only be done if the :startup capability is supported by the device.

3. Security Considerations

The same considerations as for the base NETCONF Protocol [RFC4741] are valid. It is assumed that the <partial-lock> and <partial-unlock> RPCs are only allowed for an authenticated user after passing some access control mechanism.

4. IANA Considerations

The capability's URI should be registered by IANA.

5. Change Log

draft-00 Initial version

5.1. Open Issues

Shall we allow the locking of non-existent nodes? The operator might want to reserve an object or rather its key/name even if he will create the object later.

Should we include more detailed information in error results to help debug lock conflicts, e.g. the userId of the conflicting session, the XPATH expression of the conflicting session, the instanceId of the first object where the lock conflict was found?

Should we allow users to lock parts of multiple datastores (e.g. /top/routing both in the candidate and the running datastore) in one operation? This would decrease the probability of a dead-lock, but currently the (global) <lock> operation doesn't support this.

6. XML Schema for Partial Locking

The following XML Schema defines the <partial-lock> and <partial-unlock> operations:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:ietf:params:xml:ns:netconf:partial-lock:1.0"
  xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
  targetNamespace="urn:ietf:params:xml:ns:netconf:partial-lock:1.0"
  elementFormDefault="qualified" attributeFormDefault="unqualified">

  <xs:annotation>
    <xs:documentation>
        Schema defining the partial-lock and unlock operations.
     </xs:documentation>
  </xs:annotation>

  <xs:import namespace="urn:ietf:params:xml:ns:netconf:base:1.0"
    schemaLocation="urn:ietf:params:xml:ns:netconf:base:1.0"/>

  <xs:complexType name="partialLockType">
    <xs:complexContent>
      <xs:extension base="nc:rpcOperationType">
        <xs:sequence>
          <xs:element ref="nc:config-name"/>
          <xs:element name="select" type="xs:string"
                                      maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="partialUnLockType">
```

```
        <xs:complexContent>
          <xs:extension base="nc:rpcOperationType">
```



```
            <xs:sequence>
              <xs:element name="lock-id" type="xs:unsignedInt"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>

      <!-- <partial-lock> operation -->
      <xs:element name="partial-lock" type="partialLockType"
                  substitutionGroup="nc:rpcOperation"/>

      <!-- <partial-unlock> operation -->
      <xs:element name="partial-unlock" type="partialUnLockType"
                  substitutionGroup="nc:rpcOperation"/>

      <!-- reply to <partial-lock> -->
      <xs:element name="lock-id" type="xs:unsignedInt"/>

    </xs:schema>
```

7. Acknowledgements

Thanks to Andy Bierman for providing important input to this document.

8. Normative References

[RFC4741] "NETCONF Configuration Protocol", RFC 4741, December 2006.

Authors' Addresses

Balazs Lengyel
Ericsson Hungary

Email: balazs.lengyel@ericsson.com

Martin Bjorklund
Tail-f Systems

Email: mbj@tail-f.com

Full Copyright Statement

Intellectual Property

specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).