YANG Extension and Metadata Annotation for Immutable Flag

draft-ma-netmod-immutable-flag-08

Qiufang Ma (Huawei) Presenter
Qin Wu (Huawei)
Balazs Lengyel (Ericsson)
Hongwei Li (HPE)
Recap

• Protocol operations can fail for various reasons, e.g., the server might reject configuration which it internally considers immutable
  • HW configuration, built-in keys/certs, server-provided rules/policies, server capabilities, etc
• Immutability is already allowed today and has been used by multiple vendors (e.g., there must be a user account called “root”)
• It is the aim to define a single standard solution instead of multiple existing vendor-specific solutions
  • formally flag which nodes are immutable
  • A client can benefit from it by knowing beforehand when certain otherwise valid configuration requests will cause the server to return an error
  • It is merely descriptive, more like documentation
  • It is to document server existing behavior, doesn’t apply to the server without any immutable configuration
Since IETF 116

• Discussion about transactional vs. non-transactional API
  • WG consensus that only transactional APIs should be supported
  • To avoid leading to any potential non-transactional cases, the current document narrows the scope to only support system configuration
    • Immutable configuration can only be created/updated/deleted by the server
    • Regardless of the implementation of the system configuration datastore

• Desire to allow the client to fully control the configuration in <running>
  • Immutable configuration does not affect the contents of <running> by default
  • The life cycle of immutable configuration is driven by the system
    • Only appears in (/disappears from) <system> (if exists) and <operational> by default
    • Can only be updated or deleted when software upgrades or hardware resources/license change
  • A client can merely make immutable configuration visible/invisible in rw datastores (e.g., <running>)
Since IETF 116 (cont.)

• What remains unchanged?
  • Document existing server behavior for interoperability
    • Non-transactional cases are beyond the scope
  • Server behavior on immutability is discouraged
    • Should be avoided wherever possible
  • Combined ways of a YANG extension and a metadata annotation both called “immutable” to express the behavior
    • YANG extension can be useful to convey immutability at implementation time
    • A metadata annotation is needed when a list contains both immutable server created and mutable client defined entries
  • Immutability is recursively inherited by descendant nodes, but resettable as needed
High-level Document Updates

• Define the term “immutable flag”
  • A read-only state value the server provides to describe data it considers immutable.

• Add a new section “use of ‘immutable’ Flag for different statements”
  • leaf, leaf-list, container, list, anydata, anyxml

• Define a parameter named “with-immutable” for retrieval operations
  • “immutable” annotation is not included in a response unless a client explicitly requests it with this parameter
  • Avoid breaking the legacy client which does not understand the “immutable” metadata annotation

• Remove UC4 - Declaring System defined configuration unchangeable

• Remove UC5 – Immutable BGP peer type

• Add a new UC – Declaring immutable system configuration from an LNE’s perspective
Comments, Questions, Concerns?