< draft-ietf-i2rs-architecture-07.txt

Network Working Group A. Atlas
Internet-Draft Juniper Networks
Intended status: Informational J. Halpern
Expires: June 14, 2015 Ericsson
S. Hares
Huawei
D. Ward
Cisco Systems
T. Nadeau
Brocade
December 11, 2014

An Architecture for the Interface to the Routing System
draft-ietf-i2rs-architecture-07

Abstract

This document describes an architecture for a standard, programmatic interface for state transfer in and out of the internet routing system. It describes the basic architecture, the components, and their interfaces with particular focus on those to be standardized as part of I2RS.

Table of Contents

draft-ietf-i2rs-architecture-11.txt >

Network Working Group A. Atlas
Internet-Draft Juniper Networks
Intended status: Informational J. Halpern
Expires: June 18, 2016 Ericsson
S. Hares
Huawei
D. Ward
Cisco Systems
T. Nadeau
Brocade
December 16, 2015

An Architecture for the Interface to the Routing System
draft-ietf-i2rs-architecture-11

Abstract

This document describes the IETF architecture for a standard, programmatic interface for state transfer in and out of the Internet routing system. It describes the basic architecture, the components, and their interfaces with particular focus on those to be standardized as part of the Interface to Routing System (I2RS).

Table of Contents

1. Introduction

Routers that form the internet routing infrastructure maintain state at various layers of detail and function. For example, a typical router maintains a Routing Information Base (RIB), and implements routing protocols such as OSPF, ISIS, and BGP to exchange protocol state and other information about the state of the network with other routers.

Routers convert all of this information into forwarding entries which are then used to forward packets and flows between network elements. The forwarding plane and the specified forwarding entries then contain active state information that describes the expected and observed operational behavior of the router and which is also needed by the network applications. Network-oriented applications require easy access to this information to learn the network topology, to verify that programmed state is installed in the forwarding plane, to measure the behavior of various flows, routes or forwarding entries,

skipping to change at *page 3, line 49*

This document sets out an architecture for a common, standards-based interface to this information. This Interface to the Routing System (I2RS) facilitates control and observation of the routing-related state (for example, a Routing Element RIB manager's state), as well as enabling network-oriented applications to be built on top of today's routed networks. The I2RS is a programmatic asynchronous interface for transferring state into and out of the internet routing system. This I2RS architecture recognizes that the routing system and a router's OS provide useful mechanisms that applications could harness to accomplish application-level goals.

Fundamental to the I2RS are clear data models that define the semantics of the information that can be written and read. The I2RS provides a framework for registering for and requesting the appropriate information for each particular application. The I2RS provides a way for applications to customize network behavior while leveraging the existing routing system as desired.

Although the I2RS architecture is general enough to support information and data models for a variety of data, and aspects of the I2RS solution may be useful in domain other than routing, I2RS and this document are specifically focused on an interface for routing data.

1.1. Drivers for the I2RS Architecture

There are four key drivers that shape the I2RS architecture. First is the need for an interface that is programmatic, asynchronous, and offers fast, interactive access for atomic operations. Second is the access to structured information and state that is frequently not directly configurable or modeled in existing implementations or

skipping to change at *page 4, line 38*

extensibility and provide standard data-models to be used by network applications.

I2RS is described as an asynchronous programmatic interface, the key properties of which are described in Section 5 of [I-D.ietf-i2rs-problem-statement].

The I2RS architecture facilitates obtaining information from the router. The I2RS architecture provides the ability to not only read specific information, but also to subscribe to targeted information streams and filtered and thresholded events.

Such an interface also facilitates the injection of ephemeral state into the routing system. A non-routing protocol or application could

1. Introduction

Routers that form the internet routing infrastructure maintain state at various layers of detail and function. For example, a typical router maintains a Routing Information Base (RIB), and implements routing protocols such as OSPF, IS-IS, and BGP to exchange reachability information, topology information, protocol state, and other information about the state of the network with other routers.

Routers convert all of this information into forwarding entries which are then used to forward packets and flows between network elements. The forwarding plane and the specified forwarding entries then contain active state information that describes the expected and observed operational behavior of the router and which is also needed by the network applications. Network-oriented applications require easy access to this information to learn the network topology, to verify that programmed state is installed in the forwarding plane, to measure the behavior of various flows, routes or forwarding entries,

skipping to change at *page 3, line 49*

This document sets out an architecture for a common, standards-based interface to this information. This Interface to the Routing System (I2RS) facilitates control and observation of the routing-related state (for example, a Routing Element RIB manager's state), as well as enabling network-oriented applications to be built on top of today's routed networks. The I2RS is a programmatic asynchronous interface for transferring state into and out of the internet routing system. This I2RS architecture recognizes that the routing system and a router's OS provide useful mechanisms that applications could harness to accomplish application-level goals. These network-oriented applications can leverage the I2RS programmatic interface to create new ways of combining retrieval of internet routing data, analyzing this data, setting state within routers.

Fundamental to the I2RS are clear data models that define the semantics of the information that can be written and read. The I2RS provides a framework for registering and for requesting the appropriate information for each particular application. The I2RS provides a way for applications to customize network behavior while leveraging the existing routing system as desired.

Although the I2RS architecture is general enough to support information and data models for a variety of data, and aspects of the I2RS solution may be useful in domains other than routing, I2RS and this document are specifically focused on an interface for routing data.

1.1. Drivers for the I2RS Architecture

There are four key drivers that shape the I2RS architecture. First is the need for an interface that is programmatic, asynchronous, and offers fast, interactive access for atomic operations. Second is the access to structured information and state that is frequently not directly configurable or modeled in existing implementations or

skipping to change at *page 4, line 40*

extensibility and provide standard data-models to be used by network applications.

I2RS is described as an asynchronous programmatic interface, the key properties of which are described in Section 5 of [I-D.ietf-i2rs-problem-statement].

The I2RS architecture facilitates obtaining information from the router. The I2RS architecture provides the ability to not only read specific information, but also to subscribe to targeted information streams, filtered events, and thresholded events.

Such an interface also facilitates the injection of ephemeral state into the routing system. A non-routing protocol or application could

inject state into a routing element via the state-insertion functionality of the I2RS and that state could then be distributed in a routing or signaling protocol and/or be used locally (e.g. to program the co-located forwarding plane). I2RS will only permit modification of state that would be safe, conceptually, to modify via local configuration; no direct manipulation of protocol-internal dynamically determined data is envisioned.

1.2. Architectural Overview

Figure 1 shows the basic architecture for I2RS between applications using I2RS, their associated I2RS Clients, and I2RS Agents. Applications access I2RS services through I2RS clients. A single client can provide access to one or more applications. In the figure, Clients A and B provide access to a single application, while Client P provides access to multiple applications.

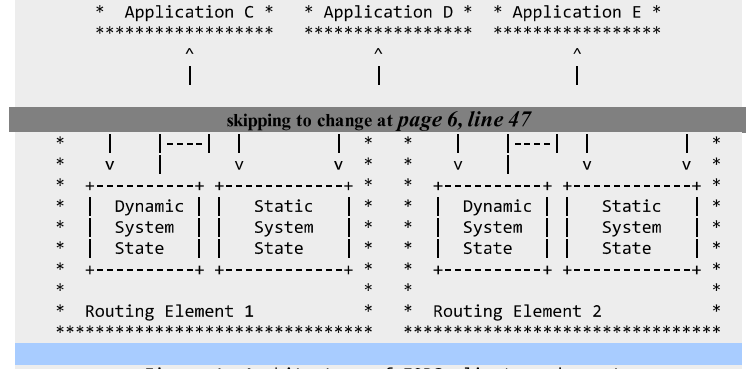
Applications can access I2RS services through local or remote clients. In the figure, Applications A and B access I2RS services through local clients, while Applications C, D and E access I2RS services through a remote client. The details of how applications communicate with a remote client is out of scope for I2RS.

An I2RS Client can access one or more I2RS agents. In the figure, Clients B and P access I2RS Agents 1 and 2. Likewise, an I2RS Agent can provide service to one or more clients. In the figure, I2RS Agent 1 provides services to Clients A, B and P while Agent 2 provides services to only Clients B and P.

I2RS agents and clients communicate with one another using an asynchronous protocol. Therefore, a single client can post multiple simultaneous requests, either to a single agent or to multiple agents. Furthermore, an agent can process multiple requests, either from a single client or from multiple clients, simultaneously.

The I2RS agent provides read and write access to selected data on the routing element that are organized into I2RS Services. Section 4 describes how access is mediated by authentication and access control mechanisms. In addition to read and write access, the I2RS agent allows clients to subscribe to different types of notifications about events affecting different object instances. An example not related to the creation, modification or deletion of an object instance is when a next-hop in the RIB is resolved enough to be used or when a particular route is selected by the RIB Manager for installation into the forwarding plane. Please see Section 7.6 and Section 7.7 for details.

The scope of I2RS is to define the interactions between the I2RS agent and the I2RS client and the associated proper behavior of the I2RS agent and I2RS client.



skipping to change at page 6, line 47

Figure 1: Architecture of I2RS clients and agents

Routing Element: A Routing Element implements some subset of the routing system. It does not need to have a forwarding plane associated with it. Examples of Routing Elements can include:

* A router with a forwarding plane and RIB Manager that runs ISIS, OSPF, BGP, PIM, etc.,
* A BGP speaker acting as a Route Reflector,
* An LSR that implements RSVP-TE, OSPF-TE, and PCEP and has a forwarding plane and associated RIB Manager,
* A server that runs ISIS, OSPF, BGP and uses ForCES to control a remote forwarding plane,

A Routing Element may be locally managed, whether via CLI, SNMP,

inject state into a routing element via the state-insertion functionality of the I2RS and that state could then be distributed in a routing or signaling protocol and/or be used locally (e.g. to program the co-located forwarding plane). I2RS will only permit modification of state that would be safe, conceptually, to modify via local configuration; no direct manipulation of protocol-internal dynamically determined data is envisioned.

1.2. Architectural Overview

Figure 1 shows the basic architecture for I2RS between applications using I2RS, their associated I2RS Clients, and I2RS Agents. Applications access I2RS services through I2RS clients. A single client can provide access to one or more applications. In the figure, Clients A and B each provide access to a single application (application A and B respectively), while Client P provides access to multiple applications.

Applications can access I2RS services through local or remote clients. In the figure, Applications A and B access I2RS services through local clients, while Applications C, D and E access I2RS services through a remote client. The details of how applications communicate with a remote client is out of scope for I2RS.

An I2RS Client can access one or more I2RS agents. In the figure 1, Clients B and P access I2RS Agents 1 and 2. Likewise, an I2RS Agent can provide service to one or more clients. In the figure, I2RS Agent 1 provides services to Clients A, B and P while Agent 2 provides services to only Clients B and P.

I2RS agents and clients communicate with one another using an asynchronous protocol. Therefore, a single client can post multiple simultaneous requests, either to a single agent or to multiple agents. Furthermore, an agent can process multiple requests, either from a single client or from multiple clients, simultaneously.

The I2RS agent provides read and write access to selected data on the routing element that are organized into I2RS Services. Section 4 describes how access is mediated by authentication and access control mechanisms. Figure 1 shows I2RS agents being able to write ephemeral static state (e.g. RIB entries), and to read from dynamic static (e.g. MPLS LSP-ID or number of active BGP peers).

In addition to read and write access, the I2RS agent allows clients to subscribe to different types of notifications about events affecting different object instances. One example of a notification of such an event (which is unrelated to an object creation, modification or deletion) is when a next-hop in the RIB is resolved enough to be used by a RIB manager for installation in the forwarding plane as part of a particular route. Please see Section 7.6 and Section 7.7 for details.

The scope of I2RS is to define the interactions between the I2RS agent and the I2RS client and the associated proper behavior of the I2RS agent and I2RS client.

skipping to change at page 7, line 4

Figure 1: Architecture of I2RS clients and agents

Routing Element: A Routing Element implements some subset of the routing system. It does not need to have a forwarding plane associated with it. Examples of Routing Elements can include:

* A router with a forwarding plane and RIB Manager that runs IS-IS, OSPF, BGP, PIM, etc.,
* A BGP speaker acting as a Route Reflector,
* An LSR that implements RSVP-TE, OSPF-TE, and PCEP and has a forwarding plane and associated RIB Manager,
* A server that runs IS-IS, OSPF, BGP and uses ForCES to control a remote forwarding plane,

A Routing Element may be locally managed, whether via CLI, SNMP,

or NETCONF.

Routing and Signaling: This block represents that portion of the Routing Element that implements part of the internet routing system. It includes not merely standardized protocols (i.e. IS-IS, OSPF, BGP, PIM, RSVP-TE, LDP, etc.), but also the RIB Manager layer.

Local Config: A Routing Element will provide the ability to configure and manage it. The Local Config may be provided via a combination of CLI, NETCONF, SNMP, etc. The black box behavior for interactions between the state that I2RS installs into the routing element and the Local Config must be defined.

Dynamic System State: An I2RS agent needs access to state on a routing element beyond what is contained in the routing subsystem. Such state may include various counters, statistics, flow data, and local events. This is the subset of operational state that is needed by network applications based on I2RS that is not contained in the routing and signaling information. How this information is provided to the I2RS agent is out of scope, but the standardized information and data models for what is exposed are part of I2RS.

skipping to change at *page 9, line 34*

An I2RS client can be seen as the part of an application that uses and supports I2RS and could be a software library.

service or I2RS Service: For the purposes of I2RS, a service refers to a set of related state access functions together with the policies that control their usage. The expectation is that a service will be represented by a data-model. For instance, 'RIB service' could be an example of a service that gives access to state held in a device's RIB.

read scope: The set of information which the I2RS client is authorized to read. The read scope specifies the access restrictions to both see the existence of data and read the value of that data.

notification scope: The set of events and associated information that the I2RS Client can request be pushed by the I2RS Agent. I2RS Clients have the ability to register for specific events and information streams, but must be constrained by the access restrictions associated with their notification scope.

write scope: The set of field values which the I2RS client is

skipping to change at *page 10, line 14*

resources: A resource is an I2RS-specific use of memory, storage, or execution that a client may consume due to its I2RS operations. The amount of each such resource that a client may consume in the context of a particular agent may be constrained based upon the client's security role. An example of such a resource could include the number of notifications registered for. These are not protocol-specific resources or network-specific resources.

role or security role: A security role specifies the scope, resources, priorities, etc. that a client or agent has.

identity: A client is associated with exactly one specific identity. State can be attributed to a particular identity. It is possible for multiple communication channels to use the same identity; in that case, the assumption is that the associated client is coordinating such communication.

secondary identity: An I2RS Client may supply a secondary opaque identity that is not interpreted by the I2RS Agent. An example use is when the I2RS Client is a go-between for multiple applications and it is necessary to track which application has requested a particular operation.

or NETCONF.

Routing and Signaling: This block represents that portion of the Routing Element that implements part of the internet routing system. It includes not merely standardized protocols (i.e. IS-IS, OSPF, BGP, PIM, RSVP-TE, LDP, etc.), but also the RIB Manager layer.

Local Configuration: A Routing Element will provide the ability to configure and manage it. The Local Configuration is defined in this architecture as the configuration parameters associated with a routing system. The Local Configuration may be read, written, or changed via a combination of CLI, NETCONF, SNMP and other protocols. The black box behavior for interactions between the ephemeral state that I2RS installs into the routing element and the Local Configuration must be defined in I2RS data models.

Dynamic System State: An I2RS agent needs access to state on a routing element beyond what is contained in the routing subsystem. Such state may include various counters, statistics, flow data, and local events. This is the subset of operational state that is needed by network applications based on I2RS that is not contained in the routing and signaling information. How this information is provided to the I2RS agent is out of scope, but the standardized information and data models for what is exposed are part of I2RS.

skipping to change at *page 9, line 41*

An I2RS client can be seen as the part of an application that uses and supports I2RS and could be a software library.

service or I2RS Service: For the purposes of I2RS, a service refers to a set of related state access functions together with the policies that control their usage. The expectation is that a service will be represented by a data-model. For instance, 'RIB service' could be an example of a service that gives access to state held in a device's RIB.

read scope: The read scope of an I2RS client within an I2RS agent is the set of information which the I2RS client is authorized to read within the I2RS agent. The read scope specifies the access restrictions to both see the existence of data and read the value of that data.

notification scope: The set of events and associated information that the I2RS Client can request be pushed by the I2RS Agent. I2RS Clients have the ability to register for specific events and information streams, but must be constrained by the access restrictions associated with their notification scope.

write scope: The set of field values which the I2RS client is

skipping to change at *page 10, line 23*

resources: A resource is an I2RS-specific use of memory, storage, or execution that a client may consume due to its I2RS operations. The amount of each such resource that a client may consume in the context of a particular agent may be constrained based upon the client's security role. An example of such a resource could include the number of notifications registered for. These are not protocol-specific resources or network-specific resources.

role or security role: A security role specifies the scope, resources, priorities, etc. that a client or agent has. Multiple identities may use the same security role. A single identity may have may have multiple roles.

identity: A client is associated with exactly one specific identity. State can be attributed to a particular identity. It is possible for multiple communication channels to use the same identity; in that case, the assumption is that the associated client is coordinating such communication.

Identity and scope: A single identity can be associated with multiple roles. Each role has its own scope and an identity associated with multiple roles can use the combined scope of all its roles. More formally, each identity has:

a read-scope that is the logical OR of the read-scopes associated with its roles,

a write-scope that is the logical OR of the write-scopes associated with its roles, and

a notification-scope that is the logical OR of the notification-scopes associated with its roles.

secondary identity: An I2RS Client may supply a secondary opaque identity that is not interpreted by the I2RS Agent. An example use is when the I2RS Client is a go-between for multiple applications and it is necessary to track which application has requested a particular operation.

Groups: NETCONF Network Access [RFC6536] refers uses the term group in terms of an Administrative group which supports support the well-established distinction between a root account and other types of less-privileged conceptual user accounts. Group still refers to a single identity (e.g. root) which is shared by a group of users.

3. Key Architectural Properties

Several key architectural properties for the I2RS protocol are elucidated below (simplicity, extensibility, and model-driven programmatic interfaces). However, some architecture principles such as performance and scaling are not described below because they are discussed in [I-D.ietf-i2rs-problem-statement] and because the performance and scaling requires varies based on the particular use-cases.

skipping to change at *page 12, line 12* / skipping to change at *page 12, line 44*

architecture and protocol(s). First, it allows for transferring data-models whose content is not explicitly implemented or understood. Second, tools can automate checking and manipulating data; this is particularly valuable for both extensibility and for the ability to easily manipulate and check proprietary data-models.

The different services provided by I2RS can correspond to separate data-models. An I2RS agent may indicate which data-models are supported.

The purpose of the data model is to provide an definition of the information regarding the routing system that can be used in operational networks. In some cases, the I2RS Working group may define an information model for a set of routing information in order to define in general terms the information for a data model. If routing information is being modeled for the first time, a logical information model may be standardized prior to creating the data model.

4. Security Considerations

This I2RS architecture describes interfaces that clearly require serious consideration of security. First, here is a brief description of the assumed security environment for I2RS. The I2RS Agent associated with a Routing Element is a trusted part of that Routing Element. For example, it may be part of a vendor-distributed signed software image for the entire Routing Element or it may be trusted signed application that an operator has installed. The I2RS Agent is assumed to have a separate authentication and authorization channel by which it can validate both the identity and permissions associated with an I2RS Client. To support numerous and speedy interactions between the I2RS Agent and I2RS Client, it is assumed that the I2RS Agent can also cache that particular I2RS Clients are trusted and their associated authorized scope. This implies that the permission information may be old either in a pull model until the I2RS Agent re-requests it, or in a push model until the authentication and authorization channel can notify the I2RS Agent of changes.

This I2RS architecture describes interfaces that clearly require serious consideration of security. As an architecture, I2RS has been designed to re-utilize existing protocols that carry network management information. Two of existing protocol which the I2RS WG has selected to attempt to re-use are NETCONF [RFC6241] and RESTCONF [I-D.ietf-netconf-restconf]. The I2RS protocol design process is to specify additional requirements which will include security for an existing protocol in order to support the I2RS architecture. After an existing protocol, e.g. NETCONF or RESTCONF, has been alter to fit the I2RS requirements, then this protocol will be reviewed to determine if it meets the I2RS security requirements.

Due to the re-use strategy of the I2RS architecture, this security section describes the assumed security environment for I2RS with additional detail on: a) identity and authentication, b) authorization, and c) client redundancy. Each protocol proposed for inclusions as an I2RS protocol will need to be evaluated for the security constraints of the protocol. The detailed requirements for the I2RS protocol and the I2RS security environment will be defined within these gloal security environments.

First, here is a brief description of the assumed security environment for I2RS. The I2RS Agent associated with a Routing Element is a trusted part of that Routing Element. For example, it may be part of a vendor-distributed signed software image for the entire Routing Element or it may be trusted signed application that an operator has installed. The I2RS Agent is assumed to have a separate authentication and authorization channel by which it can validate both the identity and permissions associated with an I2RS Client. To support numerous and speedy interactions between the I2RS Agent and I2RS Client, it is assumed that the I2RS Agent can also cache that particular I2RS Clients are trusted and their associated authorized scope. This implies that the permission information may be old either in a pull model until the I2RS Agent re-requests it, or in a push model until the authentication and authorization channel can notify the I2RS Agent of changes.

Mutual authentication between the I2RS Client and I2RS Agent is required. An I2RS Client must be able to trust that the I2RS Agent is attached to the relevant Routing Element so that write/modify operations are correctly applied and so that information received from the I2RS Agent can be trusted by the I2RS Client.

An I2RS Client is not automatically trustworthy. It has identity information and applications using that I2RS Client should be aware of the scope limitations of that I2RS Client. If the I2RS Client is

skipping to change at *page 13, line 4* / skipping to change at *page 14, line 15*

authentication and authorization for that communication is out of scope; nothing prevents I2RS and a separate authentication and authorization channel from being used. Regardless of mechanism, an

I2RS Client that is acting as a broker is responsible for determining that applications using it are trusted and permitted to make the particular requests.

Different levels of integrity, confidentiality, and replay protection are relevant for different aspects of I2RS. The primary communication channel that is used for client authentication and then used by the client to write data requires integrity, privacy and replay protection. Appropriate selection of a default required transport protocol is the preferred way of meeting these requirements.

Other communications via I2RS may not require integrity, confidentiality, and replay protection. For instance, if an I2RS Client subscribes to an information stream of prefix announcements from OSPF, those may require integrity but probably not confidentiality or replay protection. Similarly, an information stream of interface statistics may not even require guaranteed delivery. In Section 7.2, more reasoning for multiple communication

skipping to change at *page 13, line 40*

of the system must be accurately attributable. In an ideal architecture, even information collection and notification should be protected; this may be subject to engineering tradeoffs during the design.

I2RS clients may be operating on behalf of other applications. While those applications' identities are not needed for authentication or authorization, each application should have a unique opaque identifier that can be provided by the I2RS client to the I2RS agent for purposes of tracking attribution of operations to support functionality such as accounting and troubleshooting.

4.2. Authorization

All operations using I2RS, both observation and manipulation, should be subject to appropriate authorization controls. Such authorization is based on the identity and assigned role of the I2RS client performing the operations and the I2RS agent in the network element.

I2RS Agents, in performing information collection and manipulation, will be acting on behalf of the I2RS clients. As such, each operation authorization will be based on the lower of the two permissions of the agent itself and of the authenticated client. The mechanism by which this authorization is applied within the device is outside of the scope of I2RS.

The appropriate or necessary level of granularity for scope can depend upon the particular I2RS Service and the implementation's granularity. An approach to a similar access control problem is defined in the NetConf Access Control Model[RFC6536]; it allows arbitrary access to be specified for a data node instance identifier while defining meaningful manipulable defaults. The ability to specify one or more groups or roles that a particular I2RS Client belongs and then define access controls in terms of those groups or roles is expected. When a client is authenticated, its group or role membership should be provided to the I2RS Agent. The set of access control rules that an I2RS Agent uses would need to be either provided via Local Config, exposed as an I2RS Service for manipulation by authorized clients, or via some other method.

5. Network Applications and I2RS Client

I2RS is expected to be used by network-oriented applications in different architectures. While the interface between a network-oriented application and the I2RS client is outside the scope of

I2RS Client that is acting as a broker is responsible for determining that applications using it are trusted and permitted to make the particular requests.

Different levels of integrity, confidentiality, and replay protection are relevant for different aspects of I2RS. The primary communication channel that is used for client authentication and then used by the client to write data requires integrity, confidentiality and replay protection. Appropriate selection of a default required transport protocol is the preferred way of meeting these requirements.

Other communications via I2RS may not require integrity, confidentiality, and replay protection. For instance, if an I2RS Client subscribes to an information stream of prefix announcements from OSPF, those may require integrity but probably not confidentiality or replay protection. Similarly, an information stream of interface statistics may not even require guaranteed delivery. In Section 7.2, more reasoning for multiple communication

skipping to change at *page 14, line 51*

of the system must be accurately attributable. In an ideal architecture, even information collection and notification should be protected; this may be subject to engineering tradeoffs during the design.

I2RS clients may be operating on behalf of other applications. While those applications' identities are not needed for authentication or authorization, each application should have a unique opaque identifier that can be provided by the I2RS client to the I2RS agent for purposes of tracking attribution of operations to support functionality such as troubleshooting and logging of network changes.

4.2. Authorization

All operations using I2RS, both observation and manipulation, should be subject to appropriate authorization controls. Such authorization is based on the identity and assigned role of the I2RS client performing the operations and the I2RS agent in the network element. Multiple Identities may use the same role. An identity may utilize several roles. For example, an I2RS client identity may use an MPLS-Monitor role to have a read-scope that includes created MPLS LSPs and also use a Troubleshooting role to have write access to trigger active OAM such as LSP-ping.

I2RS Agents, in performing information collection and manipulation, will be acting on behalf of the I2RS clients. As such, each operation authorization will be based on the lower of the two permissions of the agent itself and of the authenticated client. The mechanism by which this authorization is applied within the device is outside of the scope of I2RS.

The appropriate or necessary level of granularity for scope can depend upon the particular I2RS Service and the implementation's granularity. An approach to a similar access control problem is defined in the NetConf Access Control Model (NACM) [RFC6536]; it allows arbitrary access to be specified for a data node instance identifier while defining meaningful manipulable defaults. The identity within NACM [RFC6536] can be specify as either a user name or a group user name (e.g. Root), and this name is linked a scope policy that contained in a set of access control rules. Similarly, it is expected the I2RS identity links to one role which has a scope policy specified by a set of access control rules. This scope policy is can be provided via Local Configuration, exposed as an I2RS Service for manipulation by authorized clients, or via some other method (e.g. AAA service)

When an I2RS client is authenticated, its identity is provided to the I2RS Agent, and this identity links to a role which links to the scope policy. Multiple identities may belong to the same role; for example, such a role might be an Internal-Routes-Monitor that allows reading of the portion of the I2RS RIB associated with IP prefixes used for internal device addresses in the AS."

4.3. Client Redundancy

I2RS must support client redundancy. At the simplest, this can be handled by having a primary and a backup network application that both use the same client identity and can successfully authenticate as such. Since I2RS does not require a continuous transport connection and supports multiple transport sessions, this can provide some basic redundancy. However, it does not address the need for troubleshooting and logging of network changes to be informed about which network application is actually active. At a minimum, basic transport information about each connection and time can be logged with the identity.

5. Network Applications and I2RS Client

I2RS is expected to be used by network-oriented applications in different architectures. While the interface between a network-oriented application and the I2RS client is outside the scope of

I2RS, considering the different architectures is important to sufficiently specify I2RS.

In the simplest architecture, a network-oriented application has an I2RS client as a library or driver for communication with routing elements.

In the broker architecture, multiple network-oriented applications communicate in an unspecified fashion to a broker application that contains an I2RS Client. That broker application requires additional functionality for authentication and authorization of the network-oriented applications; such functionality is out of scope for I2RS but similar considerations to those described in Section 4.2 do apply. As discussed in Section 4.1, the broker I2RS Client should determine distinct opaque identifiers for each network-oriented application that is using it. The broker I2RS Client can pass along

skipping to change at *page 16, line 6*

collecting and delivering large amounts of data from various parts of the routing element. Those parts may or may not actually be part of a single physical device. Thus, for scalability and robustness, it is important that the architecture allow for a distributed set of reporting components providing collected data from the I2RS agent back to the relevant I2RS clients. There may be multiple I2RS Agents within the same router. In such a case, they must have non-overlapping sets of information which they manipulate.

To facilitate operations, deployment and troubleshooting, it is important that traceability of the I2RS Agent's requests and actions be supported via a common data model.

6.2. I2RS State Storage

State modification requests are sent to the I2RS agent in a routing element by I2RS clients. The I2RS agent is responsible for applying these changes to the system, subject to the authorization discussed above. The I2RS agent will retain knowledge of the changes it has applied, and the client on whose behalf it applied the changes. The I2RS agent will also store active subscriptions. These sets of data form the I2RS data store. This data is retained by the agent until

skipping to change at *page 18, line 5*

will not store multiple alternative states, nor try to determine which one among such a plurality it should fall back to. Thus, the model followed is not like the RIB, where multiple routes are stored at different preferences. (For I2RS state in the presence of two I2RS clients, please see section 1.2 and section 7.8)

An I2RS Client may register for notifications, subject to its notification scope, regarding state modification or removal by a particular I2RS Client.

6.3. Interactions with Local Config

Changes may originate from either Local Config or from I2RS. The modifications and data stored by I2RS are separate from the local device configuration, but conflicts between the two must be resolved in a deterministic manner that respects operator-applied policy. That policy can determine whether Local Config overrides a particular I2RS client's request or vice versa. To achieve this end, either by default Local Config always wins or, optionally, a routing element may permit a priority to be configured on the device for the Local Config mechanism. The policy mechanism in the later case is comparing the I2RS client's priority with that priority assigned to the Local Config.

When the Local Config always wins, some communication between that subsystem and the I2RS Agent is still necessary. That communication contains the details of each specific device configuration change that the I2RS Agent is permitted to modify. In addition, when the system determines, that a client's I2RS state is preempted, the I2RS agent must notify the affected I2RS clients; how the system determines this is implementation-dependent.

It is critical that policy based upon the source is used because the resolution cannot be time-based. Simply allowing the most recent state to prevail could cause race conditions where the final state is not repeatably deterministic.

6.4. Routing Components and Associated I2RS Services

For simplicity, each logical protocol or set of functionality that can be compactly described in a separable information and data model is considered as a separate I2RS Service. A routing element need not implement all routing components described nor provide the associated I2RS services. When a full implementation is not mandatory, an I2RS Service should include a capability model so that implementations can indicate which parts of the service are supported. Each I2RS Service requires an information model that describes at least the following: data that can be read, data that can be written, notifications that can be subscribed to, and the capability model mentioned above.

I2RS, considering the different architectures is important to sufficiently specify I2RS.

In the simplest architecture of direct access, a network-oriented application has an I2RS client as a library or driver for communication with routing elements.

In the broker architecture, multiple network-oriented applications communicate in an unspecified fashion to a broker application that contains an I2RS Client. That broker application requires additional functionality for authentication and authorization of the network-oriented applications; such functionality is out of scope for I2RS but similar considerations to those described in Section 4.2 do apply. As discussed in Section 4.1, the broker I2RS Client should determine distinct opaque identifiers for each network-oriented application that is using it. The broker I2RS Client can pass along

skipping to change at *page 17, line 40*

collecting and delivering large amounts of data from various parts of the routing element. Those parts may or may not actually be part of a single physical device. Thus, for scalability and robustness, it is important that the architecture allow for a distributed set of reporting components providing collected data from the I2RS agent back to the relevant I2RS clients. There may be multiple I2RS Agents within the same router. In such a case, they must have non-overlapping sets of information which they manipulate.

To facilitate operations, deployment and troubleshooting, it is important that traceability of the requests received by I2RS Agent's and actions taken be supported via a common data model.

6.2. I2RS State Storage

State modification requests are sent to the I2RS agent in a routing element by I2RS clients. The I2RS agent is responsible for applying these changes to the system, subject to the authorization discussed above. The I2RS agent will retain knowledge of the changes it has applied, and the client on whose behalf it applied the changes. The I2RS agent will also store active subscriptions. These sets of data form the I2RS data store. This data is retained by the agent until

skipping to change at *page 19, line 36*

will not store multiple alternative states, nor try to determine which one among such a plurality it should fall back to. Thus, the model followed is not like the RIB, where multiple routes are stored at different preferences. (For I2RS state in the presence of two I2RS clients, please see section 1.2 and section 7.8)

An I2RS Client may register for notifications, subject to its notification scope, regarding state modification or removal by a particular I2RS Client.

6.3. Interactions with Local Configuration

Changes may originate from either Local Configuration or from I2RS. The modifications and data stored by I2RS are separate from the local device configuration, but conflicts between the two must be resolved in a deterministic manner that respects operator-applied policy. That policy can determine whether Local Configuration overrides a particular I2RS client's request or vice versa. To achieve this end, either by default Local Configuration always wins or, optionally, a routing element may permit a priority to be configured on the device for the Local Configuration mechanism. The policy mechanism in the later case is comparing the I2RS client's priority with that priority assigned to the Local Configuration.

When the Local Configuration always wins, some communication between that subsystem and the I2RS Agent is still necessary. That communication contains the details of each specific device configuration change that the I2RS Agent is permitted to modify. In addition, when the system determines, that a client's I2RS state is preempted, the I2RS agent must notify the affected I2RS clients; how the system determines this is implementation-dependent.

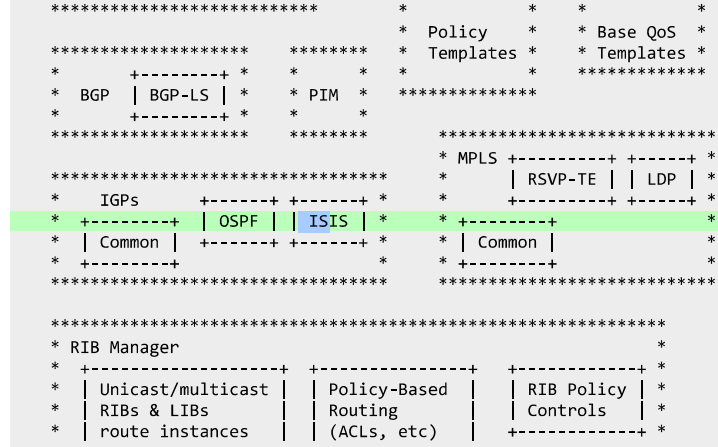
It is critical that policy based upon the source is used because the resolution cannot be time-based. Simply allowing the most recent state to prevail could cause race conditions where the final state is not repeatably deterministic.

6.4. Routing Components and Associated I2RS Services

For simplicity, each logical protocol or set of functionality that can be compactly described in a separable information and data model is considered as a separate I2RS Service. A routing element need not implement all routing components described nor provide the associated I2RS services. I2RS Services should include a capability model so that peers can determine which parts of the service are supported. Each I2RS Service requires an information model that describes at least the following: data that can be read, data that can be written, notifications that can be subscribed to, and the capability model mentioned above.

The initial services included in the I2RS architecture are as follows.

skipping to change at *page 19, line 22*



skipping to change at *page 19, line 52*

functionality (e.g. pre-defined templates to be applied to interfaces or for QoS behaviors that traffic is direct into). Section 6.4.5 discusses information modeling constructs and the range of relationship types that are applicable.

6.4.1. Routing and Label Information Bases

Routing elements may maintain one or more Information Bases. Examples include Routing Information Bases such as IPv4/IPv6 Unicast or IPv4/IPv6 Multicast. Another such example includes the MPLS Label Information Bases, per-platform or per-interface. This functionality, exposed via an I2RS Service, must interact smoothly with the same mechanisms that the routing element already uses to handle RIB input from multiple sources, so as to safely change the system state. Conceptually, this can be handled by having the I2RS Agent communicate with a RIB Manager as a separate routing source.

The point-to-multipoint state added to the RIB does not need to match to well-known multicast protocol installed state. The I2RS Agent can create arbitrary replication state in the RIB, subject to the advertised capabilities of the routing element.

6.4.2. IGPs, BGP and Multicast Protocols

A separate I2RS Service can expose each routing protocol on the device. Such I2RS services may include a number of different kinds

skipping to change at *page 22, line 30*

Information model should provide information that:

- Is X required for the data field to be accepted and applied?
- If X is optional, then how does "X" as an optional portion of data field interact with the required aspects of the data field?
- Does the data field have defaults for the mandatory portion of the field and the optional portions of the field
- Is X required to be within a particular set of values (E.g. range, length of strings)?

The information model needs to be clear about what read or write values are set by client and what responses or actions are required by the agent. It is important to indicate what is required or optional in client values and agent responses/actions.

6.4.5.3. Managing Variation: Templating

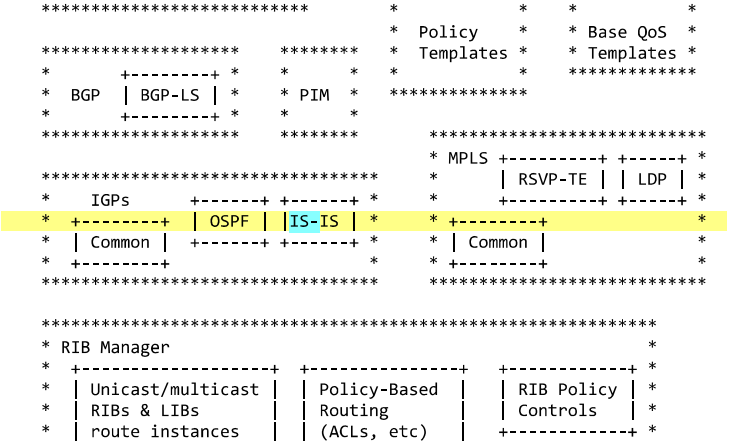
A template is a collection of information to address a problem; it

skipping to change at *page 24, line 28*

unit), and link MTU changes need to immediately propagate to the Tunnel MTU, then the tunnel is actively coupled to the link interface. This kind of active state coupling implies some sort of internal bookkeeping to ensure consistency, often conceptualized as a subscription model across objects.

The initial services included in the I2RS architecture are as follows.

skipping to change at *page 21, line 22*



skipping to change at *page 21, line 52*

functionality (e.g. pre-defined templates to be applied to interfaces or for QoS behaviors that traffic is direct into). Section 6.4.5 discusses information modeling constructs and the range of relationship types that are applicable.

6.4.1. Routing and Label Information Bases

Routing elements may maintain one or more Information Bases. Examples include Routing Information Bases such as IPv4/IPv6 Unicast or IPv4/IPv6 Multicast. Another such example includes the MPLS Label Information Bases, per-platform or per-interface or per-context.

This functionality, exposed via an I2RS Service, must interact smoothly with the same mechanisms that the routing element already uses to handle RIB input from multiple sources, so as to safely change the system state. Conceptually, this can be handled by having the I2RS Agent communicate with a RIB Manager as a separate routing source.

The point-to-multipoint state added to the RIB does not need to match to well-known multicast protocol installed state. The I2RS Agent can create arbitrary replication state in the RIB, subject to the advertised capabilities of the routing element.

6.4.2. IGPs, BGP and Multicast Protocols

A separate I2RS Service can expose each routing protocol on the device. Such I2RS services may include a number of different kinds

skipping to change at *page 24, line 35*

Information model should provide information that:

- Is X required for the data field to be accepted and applied?
- If X is optional, then how does "X" as an optional portion of data field interact with the required aspects of the data field?
- Does the data field have defaults for the mandatory portion of the field and the optional portions of the field
- Is X required to be within a particular set of values (e.g. range, length of strings)?

The information model needs to be clear about what read or write values are set by client and what responses or actions are required by the agent. It is important to indicate what is required or optional in client values and agent responses/actions.

6.4.5.3. Managing Variation: Templating

A template is a collection of information to address a problem; it

skipping to change at *page 26, line 28*

unit), and link MTU changes need to immediately propagate to the Tunnel MTU, then the tunnel is actively coupled to the link interface. This kind of active state coupling implies some sort of internal bookkeeping to ensure consistency, often conceptualized as a subscription model across objects.

7. I2RS Client Agent Interface

7.1. One Control and Data Exchange Protocol

As agreed by the I2RS working group, this I2RS architecture assumes that there is a single I2RS protocol for control and data exchange; that protocol will be based on NETCONF[RFC6241] and RESTCONF [I-D.ietf-netconf-restconf]. This helps meet the goal of simplicity and thereby enhances deployability. That protocol may need to use several underlying transports (TCP, SCTP (stream control transport protocol), DCCP (Datagram Congestion Control Protocol)), with suitable authentication and integrity protection mechanisms. These different transports can support different types of communication (e.g. control, reading, notifications, and information collection) and different sets of data. Whatever transport is used for the data exchange, it must also support suitable congestion control mechanisms. The transports chosen should be operator and implementor friendly to ease adoption.

7. I2RS Client Agent Interface

7.1. One Control and Data Exchange Protocol

This I2RS architecture assumes a data-model driven protocol where the data-models are defined in Yang 1.1 ([RFC6020]), Yang 1.1 ([I-D.ietf-netmod-rfc6020bis]), and associated Yang based model drafts ([RFC6991], [RFC7223], [RFC7224], [RFC7277], [RFC7317]). Two the protocols to be expanded to support the I2RS protocol are NETCONF [RFC6241] and RESTCONF [I-D.ietf-netconf-restconf]. This helps meet the goal of simplicity and thereby enhances deployability. The I2RS protocol may need to use several underlying transports (TCP, SCTP (stream control transport protocol), DCCP (Datagram Congestion Control Protocol)), with suitable authentication and integrity protection mechanisms. These different transports can support different types of communication (e.g. control, reading, notifications, and information collection) and different sets of data. Whatever transport is used for the data exchange, it must also support suitable congestion control mechanisms. The transports chosen should be operator and implementor friendly to ease adoption.

7.2. Communication Channels

Multiple communication channels and multiple types of communication channels are required. There may be a range of requirements (e.g. confidentiality, reliability), and to support the scaling there may need to be channels originating from multiple sub-components of a routing element and/or to multiple parts of an I2RS client. All such communication channels will use the same higher level protocol. Use of additional channels for communication will be coordinated between

skipping to change at *page 25, line 38*

Multiple communication channels and multiple types of communication channels are required. There may be a range of requirements (e.g. confidentiality, reliability), and to support the scaling there may need to be channels originating from multiple sub-components of a routing element and/or to multiple parts of an I2RS client. All such communication channels will use the same higher level protocol. Use of additional channels for communication will be coordinated between

skipping to change at *page 27, line 40*

The protocol capability negotiation can be segmented into the basic version negotiation (required to ensure basic communication), and the more complex capability exchange which can take place within the base protocol mechanisms. In particular, the more complex protocol and mechanism negotiation can be addressed by defining information models for both the I2RS Agent and the I2RS Client. These information models can describe the various capability options. This can then represent and be used to communicate important information about the agent, and the capabilities thereof.

7.4. Identity and Security Role

Each I2RS Client will have a unique identity; it can also have secondary identities to be used for troubleshooting. A secondary identity is merely a unique, opaque identifier that may be helpful in troubleshooting. Via authentication and authorization mechanisms based on the primary unique identity, the I2RS Client will have a specific scope for reading data, for writing data, and limitations on the resources that can be consumed. The scopes need to specify both the data and the value ranges.

7.4.1. Client Redundancy

I2RS must support client redundancy. At the simplest, this can be handled by having a primary and a backup network application that both use the same client identity and can successfully authenticate as such. Since I2RS does not require a continuous transport connection and supports multiple transport sessions, this can provide some basic redundancy. However, it does not address concerns for troubleshooting and accountability about knowing which network application is actually active. At a minimum, basic transport information about each connection and time can be logged with the identity.

7.4. Scope Policy Specifications

As section 4.1 and 4.2 describe, each I2RS Client will have a unique identity and it may have a secondary identity (see section 2) to aid in troubleshooting. As section 4 indicates, all authentication and authorization mechanisms are based on the primary Identity which links to a role with scope policy for reading data, for writing data, and limitations on the resources that can be consumed. Specifications for scope policy need to specify the data and value ranges for portion of scope policy.

7.5. Connectivity

A client may or may not maintain an active communication channel with an agent. Therefore, an agent may need to open a communication channel to the client to communicate previously requested information. The lack of an active communication channel does not imply that the associated client is non-functional. When communication is required, the agent or client can open a new communication channel.

skipping to change at *page 27, line 32*

specific types of events and filtering operations can vary by information model and need to be specified as part of the information model.

The I2RS information model needs to include representation of these events. As discussed earlier, the capability information in the model will allow I2RS clients to understand which events a given I2RS Agent is capable of generating.

For performance and scaling by the I2RS client and general information privacy, an I2RS Client needs to be able to register for just the events it is interested in. It is also possible that I2RS might might provide a stream of notifications via a publish/subscribe mechanism that is not amenable to having the I2RS agent do the

skipping to change at *page 29, line 16*

specific types of events and filtering operations can vary by information model and need to be specified as part of the information model.

The I2RS information model needs to include representation of these events. As discussed earlier, the capability information in the model will allow I2RS clients to understand which events a given I2RS Agent is capable of generating.

For performance and scaling by the I2RS client and general information confidentiality, an I2RS Client needs to be able to register for just the events it is interested in. It is also possible that I2RS might provide a stream of notifications via a publish/subscribe mechanism that is not amenable to having the I2RS

filtering.

7.7. Information collection

One of the other important aspects of the I2RS is that it is intended to simplify collecting information about the state of network elements. This includes both getting a snapshot of a large amount of data about the current state of the network element, and subscribing to a feed of the ongoing changes to the set of data or a subset thereof. This is considered architecturally separate from notifications due to the differences in information rate and total

skipping to change at *page 29, line 45*

resources, whether those be operations in a time-frame, entries in the RIB, stored operations to be triggered, etc. The ability to set resource limits based upon authorization is important.

Configuration Interactions: The interaction of state installed via the I2RS and via a router's configuration needs to be clearly defined. As described in this architecture, a simple priority that is configured is used to provide sufficient policy flexibility.

9. IANA Considerations

This document includes no request to IANA.

10. Acknowledgements

Significant portions of this draft came from draft-ward-i2rs-framework-00 and draft-atlas-i2rs-policy-framework-00.

The authors would like to thank Nitin Bahadur, Shane Amante, Ed

skipping to change at *page 30, line 23*

Brim, Thomas Narten, Dean Bogdanovi, Tom Petch, Robert Raszuk, Sriganesh Kini, John Mattsson, Nancy Cam-Winget, DaCheng Zhang, Qin Wu, Ahmed Abro, Salman Asadullah, and Eric Yu. for their suggestions and review.

11. Informative References

[I-D.ietf-i2rs-problem-statement]
Atlas, A., Nadeau, T., and D. Ward, "Interface to the Routing System Problem Statement", draft-ietf-i2rs-problem-statement-04 (work in progress), June 2014.

[I-D.ietf-idr-ls-distribution]
Gredler, H., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and TE Information using BGP", draft-ietf-idr-ls-distribution-07 (work in progress), November 2014.

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-03 (work in progress), October 2014.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

agent do the filtering.

7.7. Information collection

One of the other important aspects of the I2RS is that it is intended to simplify collecting information about the state of network elements. This includes both getting a snapshot of a large amount of data about the current state of the network element, and subscribing to a feed of the ongoing changes to the set of data or a subset thereof. This is considered architecturally separate from notifications due to the differences in information rate and total

skipping to change at *page 31, line 25*

resources, whether those be operations in a time-frame, entries in the RIB, stored operations to be triggered, etc. The ability to set resource limits based upon authorization is important.

Configuration Interactions: The interaction of state installed via the I2RS and via a router's configuration needs to be clearly defined. As described in this architecture, a simple priority that is configured is used to provide sufficient policy flexibility.

Traceability of Interactions: The ability to trace the interactions of the requests received by the I2RS Agent's and actions taken by the I2RS agents is needed so that operations can monitor I2RS Agents during deployment, and troubleshoot software or network problems.

Notification Subscription Service: The ability for an I2RS Client to subscribe to a notification stream pushed from the I2RS Agent (rather than having I2RS client poll the I2RS agent) provides a more scalable notification handling for the I2RS Agent-Client interactions.

9. IANA Considerations

This document includes no request to IANA.

10. Acknowledgements

Significant portions of this draft came from draft-ward-i2rs-framework-00 and draft-atlas-i2rs-policy-framework-00.

The authors would like to thank Nitin Bahadur, Shane Amante, Ed

skipping to change at *page 32, line 12*

Brim, Thomas Narten, Dean Bogdanovi, Tom Petch, Robert Raszuk, Sriganesh Kini, John Mattsson, Nancy Cam-Winget, DaCheng Zhang, Qin Wu, Ahmed Abro, Salman Asadullah, and Eric Yu. for their suggestions and review.

11. Informative References

[I-D.ietf-i2rs-problem-statement]
Atlas, A., Nadeau, T., and D. Ward, "Interface to the Routing System Problem Statement", draft-ietf-i2rs-problem-statement-06 (work in progress), January 2015.

[I-D.ietf-idr-ls-distribution]
Gredler, H., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and TE Information using BGP", draft-ietf-idr-ls-distribution-13 (work in progress), October 2015.

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-09 (work in progress), December 2015.

[I-D.ietf-netmod-rfc6020bis]
Bjorklund, M., "The YANG 1.1 Data Modeling Language", draft-ietf-netmod-rfc6020bis-09 (work in progress), December 2015.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://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, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013,

<http://www.rfc-editor.org/info/rfc6991>.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <http://www.rfc-editor.org/info/rfc7223>.

[RFC7224] Bjorklund, M., "IANA Interface Type YANG Module", RFC 7224, DOI 10.17487/RFC7224, May 2014, <http://www.rfc-editor.org/info/rfc7224>.

[RFC7277] Bjorklund, M., "A YANG Data Model for IP Management", RFC 7277, DOI 10.17487/RFC7277, June 2014, <http://www.rfc-editor.org/info/rfc7277>.

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

Authors' Addresses

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Authors' Addresses

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

End of changes. 52 change blocks.

202 lines changed or deleted ***325 lines changed or added***