I2RS
Internet-Draft
Intended status: Informational
Expires: December 9, 2014

J. Clarke
G. Salgueiro
C. Pignataro
Cisco

June 7, 2014

Interface to the Routing System (I2RS) Traceability: Framework and Information Model
draft-clarke-i2rs-traceability-02

Abstract

This document describes a framework for traceability in the Interface to the Routing System (I2RS) and information model for that framework. It specifies the motivation, requirements, use cases, and defines an information model for recording interactions between elements implementing the I2RS protocol. This framework provides a consistent tracing interface for components implementing the I2RS architecture to record what was done, by which component, and when. It aims to improve the management of I2RS implementations, and can be used for troubleshooting, auditing, forensics, and accounting purposes.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 9, 2014.

Table of Contents

1. Introduction

The architecture for the Interface to the Routing System ([I-D.ietf-i2rs-architecture]) specifies that I2RS Clients wishing to retrieve or change routing state on a routing element MUST authenticate to an I2RS Agent. The I2RS Client will have a unique identity it provides for authentication, and should provide another, opaque identifier for applications (or actors) communicating through it. The programming of routing state will produce a return code

containing the results of the specified operation and associated reason(s) for the result. All of this is critical information to be used for understanding the history of I2RS interactions.

This document describes use cases for I2RS traceability. Based on these use cases, the document proposes an information model and reporting requirements to provide for effective recording of I2RS interactions. In this context, effective troubleshooting means being able to identify what operation was performed by a specific I2RS Client, what was the result of the operation, and when that operation was performed.

Discussions about the retention of the data logged as part of I2RS traceability, while important, are outside of the scope of this document.

2. Terminology and Conventions

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 [RFC2119].

The architecture specification for I2RS [I-D.ietf-i2rs-architecture] defines additional terms used in this document that are specific to the I2RS domain, such as "I2RS Agent", "I2RS Client", etc. The reader is expected to be familiar with the terminology and concepts defined in [I-D.ietf-i2rs-architecture].

The IP addresses used in the example in this document correspond to the documentation address blocks 192.0.2.0/24 (TEST-NET-1), 198.51.100.0/24 (TEST-NET-2) and 203.0.113.0/24 (TEST-NET-3) as described in [RFC5737].

3. Motivation

As networks scale and policy becomes an increasingly important part of the control plane that creates and maintains the forwarding state, operational complexity increases as well. I2RS offers more granular and coherent control over policy and control plane state, but it also removes or reduces the locality of the policy that has been applied to the control plane at any individual forwarding device. The ability to automate and abstract even complex policy-based controls highlights the need for an equally scalable traceability function to provide event-level granularity of the routing system compliant with the requirements of I2RS (Section 5 of [I-D.ietf-i2rs-problem-statement]).

4. Use Cases

An obvious motivation for I2RS traceability is the need to troubleshoot and identify root-causes of problems in these increasingly complex routing systems. For example, since I2RS is a high-throughput multi-channel, full duplex and highly responsive interface, I2RS Clients may be performing a large number of operations on I2RS Agents concurrently or at nearly the same time and quite possibly in very rapid succession. As these many changes are made, the network reacts accordingly. These changes might lead to a race condition, performance issues, data loss, or disruption of services. In order to isolate the root cause of these issues it is critical that a network operator or administrator has visibility into what changes were made via I2RS at a specific time.

Some network environments have strong auditing requirements for configuration and runtime changes. Other environments have policies that require saving logging information for operational or regulatory compliance considerations. These requirements therefore demand that I2RS provides an account of changes made to network element routing systems.

As I2RS becomes increasingly pervasive in routing environments, a traceability model offers significant advantages and facilitates the following use cases:

- Automated event correlation, trend analysis, and anomaly detection.
- Trace log storage for offline (manual or tools) analysis.
- Improved accounting of routing system transactions.
- Standardized structured data format for writing common tools.
- Common reference for automated testing and incident reporting.
- Real-time monitoring and troubleshooting.
- Enhanced network audit, management and forensic analysis capabilities.

5. Information Model

5.1. I2RS Traceability Framework

This section describes a framework for I2RS traceability based on the I2RS Architecture. Some notable elements on the architecture are highlighted herein.

The interaction between the optional northbound actor, I2RS Client, I2RS Agent, the Routing System and the data captured in the I2RS trace log is shown in Figure 1.

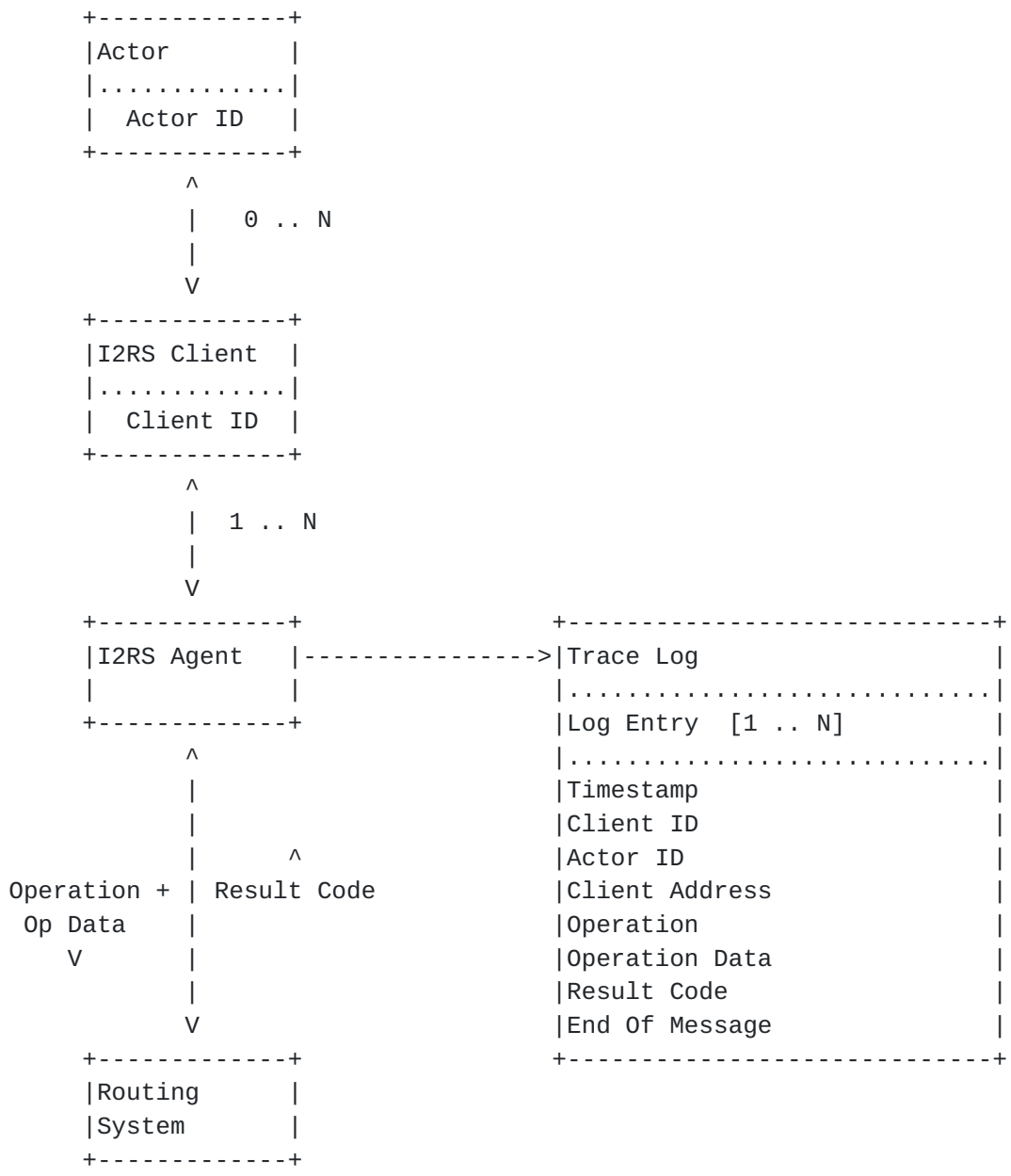

Figure 1: I2RS Interaction Trace Log Capture

5.2. I2RS Trace Log Mandatory Fields

In order to ensure that each I2RS interaction can be properly traced back to the Client that made the request at a specific point in time, the following information MUST be collected and stored by the Agent.

The list below describes the fields captured in the I2RS trace log.

Entry ID: This is a unique identifier for each entry in the I2RS trace log. Since multiple operations can occur from the same

client at the same time, it is important to have an identifier that can be unambiguously associated to a specific entry.

Timestamp: The specific time, adhering to [RFC3339] format, at which the I2RS transaction occurred. Given that many I2RS transactions can occur in rapid succession, the use of fractional seconds MUST be used to provide adequate granularity.

Client Identifier: The I2RS Client identifier used to authenticate the Client to the I2RS Agent.

Actor Identifier: This is an opaque identifier that may be known to the Client from a northbound controlling application. This is used to trace the northbound actor driving the actions of the Client. The Client may not provide this identifier to the Agent if there is no external actor driving the Client. However, this field MUST be logged. If the Client does not provide an actor ID, then the Agent MUST log an UNAVAILABLE value in the field.

Client Address: This is the network address of the client that connected to the Agent. For example, this may be an IPv4 or IPv6 address. [Note: will I2RS support interactions that have no network address? If so this field will need to be updated.]

Operation: This is the I2RS operation performed. For example, this may be an add route operation if a route is being inserted into a routing table.

Operation Data: This field comprises the data passed to the Agent to complete the desired operation. For example, if the operation is a route add operation, the Operation Data would include the route prefix, prefix length, and next hop information to be inserted as well as the specific routing table to which the route will be added. The operation data can also include interface information. Some operations may not provide operation data, and in those cases this field MUST be logged as a NULL string.

Result Code: This field holds the result of the operation. In the case of RIB operations, this MUST be the return code as specified in Section 4 of [I-D.nitinb-i2rs-rib-info-model]. The operation may not complete with a result code in the case of a timeout. If the operation fails to complete, it MUST still log the attempted operation with an appropriate result code (e.g., a result code indicating a timeout).

End Of Message: Each log entry SHOULD have an appropriate End Of Message (EOM) indicator. See section Section 5.3 below for more details.

5.3. End of Message Marker

Because of variability within I2RS trace log fields, implementors MUST use a format-appropriate end of message (EOM) indicator in order to signify the end of a particular record. That is, regardless of format, the I2RS trace log MUST provide a distinct way of distinguishing between the end of one record and the beginning of another. For example, in a linear formated log (similar to syslog) the EOM marker may be a newline character. In an XML formated log, the schema would provide for element tags that denote beginning and end of records. In a JSON formated log, the syntax would provide record separation (likely by comma-separated array elements).

5.4. I2RS Trace Log Extensibility and Optional Fields

[NOTE: This section is TBD based on further development of I2RS WG milestones.]

5.5. I2RS Trace Log Syntax

The following describes the trace log information model using the YANG modeling language [RFC6020].

5.5.1. Structure of the I2RS Trace Log

The structure of the I2RS traceability model, as later defined in the YANG module "i2rs-trace-log", is depicted in the following diagram. This tree representation illustrates the structure of I2RS Trace Log YANG model and does not depict all definitions; it is merely intended to illustrate the overall structure.

```
module: i2rs-trace-log
   +--rw i2rs-trace-log
      +--rw log-enable?   boolean
      +--ro log-entry* [log-entry-id]
         +--ro log-entry-id      log-entry-id
         +--ro timestamp         timestamp
         +--ro client-id         client-id
         +--ro actor-id          actor-id
         +--ro client-addr       client-addr
         +--ro operation         operation
         +--ro operation-data    operation-data
         +--ro result-code       result-code
```

The idea of using a UUID for the Client identifier ensures the ID is unique not just in the scope of the current I2RS Agent, but across Agents as well. This ensures that two clients that are unaware of

each other will not allocate the same Client ID. That does not preclude two Clients acting as one for purposes of high availability from sharing the same UUID as generated by one one of the Clients.

The "timestamp" field is defined in [RFC3339]. As stated in Section 5.2 the fractional second format MUST be used to provide proper granularity.

The values for "operation", "operation-data" and "result-code" are suggestions as the protocol has not been defined yet. By making these human-readable (as opposed to opcodes) the log becomes more easily consumable by operators and administrators trying to troubleshoot issues relating to I2RS. Even in cases where the operations or codes might appear as opcodes on the wire, their textual translations MUST be included in the log. The opcodes themselves MAY appear in parentheses after the textual representation.

5.5.2. I2RS Trace Log Yang Module

The I2RS traceability model is defined in the following YANG module.

This YANG module imports typedefs from [RFC6021].

```
<CODE BEGINS>
file "i2rs-trace-log@2014-06-06.yang"
module i2rs-trace-log {
        yang-version 1;
        namespace "urn:TBD:params:xml:ns:yang:i2rs:trace-log";
        // This namespace should be considered with other I2RS YANG
        // models.
        prefix i2rslog;

        import ietf-yang-types {
                prefix "yang";
        }

        import ietf-inet-types {
                prefix "inet";
        }

        organization "TBD";

        contact
           "Joe Clarke jclarke@cisco.com
           Gonzalo Salgueiro gsalguei@cisco.com
           Carlos Pignataro cpignata@cisco.com";
```

```
    description
        "This module defines the model for I2RS traceability
    based on the I2RS architecture as defined in
    draft-ietf-i2rs-architecture.

    Copyright (c) 2014 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.

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

    reference
        "draft-ietf-i2rs-architecture: An Architecture for the
    Interface to the Routing System";

    revision "2014-06-03" {
            description "Initial revision";
            reference "TBD";
    }

    typedef log-entry-id {
            type uint64;
            description
                "A unique identifier for I2RS log entries.";
    }

    typedef timestamp {
            type yang:date-and-time;
            description
                "Timestamp for I2RS transactions.";
    }

    typedef client-id {
            type yang:uuid;
            description
                "The I2RS Client identifier used to authenticate
            the Client to the I2RS Agent.";
    }

    typedef actor-id {
            type union {
                    type string {
                            pattern "[^\r\n]+";
                    }
```

```
                        type enumeration {
                                enum UNAVAILABLE {
                                        description
                                            "The Actor ID was not
                                        specified by the Client.";
                                }
                        }
                }
                description
                    "Identifier used to trace the northbound actor
                driving the actions of the Client.";
        }

        typedef client-addr {
                type inet:ip-address;
                description
                    "IP address of the client that connected to the
                Agent.";
        }

        typedef operation {
                type string {
                        pattern "[a-zA-Z] [a-zA-Z_\-\(\)]*";
                    }
                description
                    "This is the I2RS operation performed.";
        }

        typedef operation-data {
                type union {
                        type string;
                        type enumeration {
                                enum NULL {
                                        description
                                            "No additional operation
                                        data was required.";
                                }
                        }
                }
                description
                    "Data passed to the Agent to complete the desired
                operation.";
        }

        typedef result-code {
                type string {
                        pattern "[a-zA-Z0-9_\-\(\)]+";
                }
```

```
                description
                    "Result code for the operation.";
            }

            container i2rs-trace-log {
                description
                    "This is the model for I2RS traceability.  An
                I2RS log is comprised of the following mandatory
                fields.  Each field MUST be identified by a unique
                log-entry-id.";
                leaf log-enable {
                        type boolean;
                        default "true";
                        description
                            "Enable/Disable I2RS logging.";
                }
                list log-entry {
                        key "log-entry-id";
                        config false;
                        description
                            "Each element in an I2RS trace log is
                        discrete and identified by its unique log
                        entry ID.";
                        leaf log-entry-id {
                                type log-entry-id;
                                config false;
                                description
                                    "This is a unique identifier for
                                each entry in the I2RS trace log.
                                Since multiple operations can occur
                                from the same client at the same
                                time, it is important to have an
                                identifier that can be unambiguously
                                associated to a specific entry.";
                        }
                        leaf timestamp {
                                type timestamp;
                                config false;
                                mandatory true;
                                description
                                    "The specific time, adhering to
                                RFC3339 format, at which the I2RS
                                transaction occurred. Given that
                                many I2RS transactions can occur in
                                rapid succession, the use of
                                fractional seconds MUST be used to
                                provide adequate granularity.";
                                reference
```

```
                "RFC 3339: Date and Time on the
            Internet: Timestamps";
        }
        leaf client-id {
            type client-id;
            config false;
            mandatory true;
            description
                "The I2RS Client identifier used
            to authenticate the Client to the
            I2RS Agent.";
        }
        leaf actor-id {
            type actor-id;
            config false;
            mandatory true;
            description
                "This is an opaque identifier
             that may be known to the Client
             from a northbound controlling
             application. This is used to
             trace the northbound actor
             driving the actions of the
             Client. The Client may not
             provide this identifier to the
             Agent if there is no external
             actor driving the Client. In
             that case, the special value,
             UNAVAILABLE is used to denote no
             Actor ID.";
        }
        leaf client-addr {
            type client-addr;
            config false;
            mandatory true;
            description
                "This is the IP address of the
            client that connected to the Agent.";
        }
        leaf operation {
            type operation;
            config false;
            mandatory true;
            description
                "This is the I2RS operation
            performed.";
        }
        leaf operation-data {
```

```
                        type operation-data;
                        config false;
                        mandatory true;
                        description
                            "This field comprises the data
                        passed to the Agent to complete the
                        desired operation.  If no additional
                        operation data is required, then this
                        field should be set to the special
                        value, NULL.";
                    }
                    leaf result-code {
                        type result-code;
                        config false;
                        mandatory true;
                        description
                            "This field holds the result of
                        the operation. In the case of RIB
                        operations, this MUST be the return
                        code as specified in Section 4 of
                        draft-nitinb-i2rs-rib-info-model.
                        The operation may not complete with
                        a result code in the case of a
                        timeout. If the operation fails to
                        complete, it MUST still log the
                        attempted operation with an
                        appropriate result code (e.g., a
                        result code indicating a timeout).";
                        reference
                        "draft-nitinb-i2rs-rib-info-model:
                        Routing Information Base Info Model";
                    }
                }
            }
        }
    }
    <CODE ENDS>
```

6. Examples

Here is a proposed sample of what the fields might look like in an I2RS trace log. This is only an early proposal. These values are subject to change.

```
Entry ID:       1
Timestamp:      2013-09-03T12:00:01.21+00:00
Client ID:      5CEF1870-0326-11E2-A21F-0800200C9A66
Actor ID:       com.example.RoutingApp
Client Address: 192.0.2.2
Operation:      ROUTE_ADD
Operation Data: PREFIX 203.0.113.0 PREFIX-LEN 24 NEXT-HOP
                198.51.100.1
Result Code:    SUCCESS(0)
```

7. Operational Guidance

Specific operational procedures regarding temporary log storage, rollover, retrieval, and access of I2RS trace logs is out of scope for this document. Organizations employing I2RS trace logging are responsible for establishing proper operational procedures that are appropriately suited to their specific requirements and operating environment. In this section we only provide fundamental and generalized operational guidelines that are implementation-independent.

7.1. Trace Log Creation

The I2RS Agent interacts with the Routing and Signaling functions of the Routing Element. Since the I2RS Agent is responsible for actually making the routing changes on the associated network device, it creates and maintains a log of transactions that can be retrieved to troubleshoot I2RS-related impact to the network.

7.2. Trace Log Temporary Storage

The trace information may be temporarily stored either in an in-memory buffer or as a file local to the Agent. Care should be given to the number of I2RS transactions expected on a given agent so that the appropriate storage medium is used and to maximize the effectiveness of the log while not impacting the performance and health of the Agent. Section 7.3 talks about rotating the trace log in order to preserve the transaction history without exhausting Agent or network device resources. It is perfectly acceptable, therefore, to use both an in-memory buffer for recent transactions while rotating or archiving older transactions to a local file.

It is outside the scope of this document to specify the implementation details (i.e., size, throughput, data protection, privacy, etc.) for the physical storage of the I2RS log file. Data retention policies of the I2RS traceability log is also outside the scope of this document.

7.3. Trace Log Rotation

In order to prevent the exhaustion of resources on the I2RS Agent or its associated network device, it is RECOMMENDED that the I2RS Agent implements trace log rotation. The details on how this is achieved are left to the implementation and outside the scope of this document. However, it should be possible to do file rotation based on either time or size of the current trace log. If file rollover is supported, multiple archived log files should be supported in order to maximize the troubleshooting and accounting benefits of the trace log.

7.4. Trace Log Retrieval

Implementors are free to provide their own, proprietary interfaces and develop custom tools to retrieve and display the I2RS trace log. These may include the display of the I2RS trace log as Command Line Interface (CLI) output. However, a key intention of defining this information model is to establish an implementor-agnostic and consistent interface to collect I2RS trace data. Correspondingly, retrieval of the data should also be made implementor-agnostic.

The following three sections describe potential ways the trace log can be accessed. At least one of these three MUST be used, with the I2RS mechanisms being preferred as they are implementor-independent approaches to retrieving the data.

7.4.1. Retrieval Via Syslog

The syslog protocol [RFC5424] is a standard way of sending event notification messages from a host to a collector. However, the protocol does not define any standard format for storing the messages, and thus implementors of I2RS tracing would be left to define their own format. So, while the data contained within the syslog message would adhere to this information model, and may be consumable by a human operator, it would not be easily parseable by a machine. Therefore, syslog MAY be employed as a means of retrieving or disseminating the I2RS trace log contents.

7.4.2. Retrieval Via I2RS Information Collection

Section 6.7 of the I2RS architecture [I-D.ietf-i2rs-architecture] defines a mechanism for information collection. The information collected includes obtaining a snapshot of a large amount of data from the network element. It is the intent of I2RS to make this data available in an implementor-agnostic fashion. Therefore, the I2RS trace log SHOULD be made available via the I2RS information

collection mechanism either as a single snapshot or via a subscription stream.

7.4.3. Retrieval Via I2RS Pub-Sub

Section 6.7 of the I2RS architecture [I-D.ietf-i2rs-architecture] goes on to define a publish-subscribe mechanism for a feed of changes happening within the I2RS layer. I2RS Agents SHOULD support publishing I2RS trace log information to that feed as described in that document. Subscribers would then receive a live stream of I2RS interactions in trace log format and could flexibly choose to do a number of things with the log messages. For example, the subscribers could log the messages to a datastore, aggregate and summarize interactions from a single client, etc. Using pub-sub for the purpose of logging I2RS interactions augments the areas described by [I-D.camwinget-i2rs-pubsub-sec]. The full range of potential activites is virtually limitless and the details of how they are performed are outside the scope of this document, however.

8. IANA Considerations

The YANG module implies a namespace that will need to be registered with IANA. However, the I2RS WG will likely request such a namespace for other work, so the registration of that namespace may occur in a separate document. This section will be updated as these WG decisions are made.

9. Security Considerations

The I2RS trace log, like any log file, reveals the state of the entity producing it as well as the identifying information elements and detailed interactions of the system containing it. The information model described in this document does not itself introduce any security issues, but it does define the set of attributes that make up an I2RS log file. These attributes may contain sensitive information and thus should adhere to the security, privacy and permission policies of the organization making use of the I2RS log file.

It is outside the scope of this document to specify how to protect the stored log file, but it is expected that adequate precautions and security best practices such as disk encryption, appropriately restrictive file/directory permissions, suitable hardening and physical security of logging entities, mutual authentication, transport encryption, channel confidentiality, and channel integrity if transferring log files. Additionally, the potentially sensitive information contained in a log file SHOULD be adequately anonymized or obfuscated by operators to ensure its privacy.

10. Acknowledgments

The authors would like to thank Alia Atlas for her initial feedback and overall support for this work. Additionally, the authors acknowledge Alvaro Retana, Russ White, Matt Birkner, Jeff Haas, Joel Halpern and Dean Bogdanovich for their reviews, contributed text, and suggested improvements to this document.

11. References

11.1. Normative References

[I-D.ietf-i2rs-architecture]
Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-00 (work in progress), August 2013.

[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-00 (work in progress), August 2013.

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

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

[RFC6021] Schoenwaelder, J., "Common YANG Data Types", RFC 6021, October 2010.

11.2. Informative References

[I-D.camwinget-i2rs-pubsub-sec]
Beck, K., Cam-Winget, N., and D. McGrew, "Using the Publish-Subscribe Model in the Interface to the Routing System", draft-camwinget-i2rs-pubsub-sec-00 (work in progress), July 2013.

[I-D.nitinb-i2rs-rib-info-model]
Bahadur, N., Folkes, R., Kini, S., and J. Medved, "Routing Information Base Info Model", draft-nitinb-i2rs-rib-info-model-02 (work in progress), August 2013.

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, March 2009.

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", RFC 5737, January 2010.

Authors' Addresses

Joe Clarke
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC 27709
US

Phone: +1-919-392-2867
Email: jclarke@cisco.com

Gonzalo Salgueiro
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC 27709
US

Email: gsalguei@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC 27709
US

Email: cpignata@cisco.com