Internet Engineering Task Force (IETF)
Request for Comments: 7922
Category: Informational
ISSN: 2070-1721
J. Clarke
G. Salgueiro
C. Pignataro
Cisco
June 2016

Interface to the Routing System (I2RS) Traceability: Framework and Information Model

Abstract

This document describes a framework for traceability in the Interface to the Routing System (I2RS) and the information model for that framework. It specifies the motivation, requirements, and 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 document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7922.

Copyright Notice

Table of Contents

1. Introduction 3
2. Terminology and Conventions 3
3. Motivation 4
4. Use Cases 4
5. Information Model 5
   5.1. I2RS Traceability Framework 5
   5.2. I2RS Trace Log Fields 7
   5.3. End of Message Marker 11
6. Examples 11
7. Operational Guidance 11
   7.1. Trace Log Creation 12
   7.2. Trace Log Temporary Storage 12
   7.3. Trace Log Rotation 13
   7.4. Trace Log Retrieval 13
        7.4.1. Retrieval via Syslog 14
        7.4.2. Retrieval via I2RS Information Collection 14
        7.4.3. Retrieval via I2RS Pub/Sub 14
8. Security Considerations 15
9. References 16
   9.1. Normative References 16
   9.2. Informative References 16
Acknowledgments 17
Authors' Addresses 17

1. Introduction

The architecture for the Interface to the Routing System [RFC7921] specifies that I2RS clients wishing to retrieve or change the 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 identity for applications 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 defines the framework necessary to trace those interactions between the I2RS client and I2RS agent. It goes on to describe use cases for traceability within I2RS. 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 via the I2RS agent, what was the result of the operation, and when that operation was performed.

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 [RFC7921] 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 [RFC7921].

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 recording at event-level granularity of the evolution of the routing system compliant with the requirements of I2RS (Section 5 of [RFC7920]).

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 that supports controllable trace log retention using a standardized structured data format offers significant advantages, such as the ability to create common tools supporting automated testing, and facilitates the following use cases:

- o real-time monitoring and troubleshooting of router events;
- o automated event correlation, trend analysis, and anomaly detection;
- o offline (manual or tools-based) analysis of router state evolution from the retained trace logs;
- o enhanced network audit, management, and forensic analysis capabilities;
- o improved accounting of routing system operations; and
- o providing a standardized format for incident reporting and test logging.

5. Information Model

These sections describe the I2RS traceability information model and the details about each of the fields to be logged.

5.1. I2RS Traceability Framework

This section describes a framework for I2RS traceability based on the I2RS Architecture.

The interaction between the optional network application that drives client activity, I2RS client, I2RS agent, the Routing System, and the data captured in the I2RS trace log is shown in Figure 1.


```
               +----------------+
          +-----------------+    |
          |Application      |    |
          |..............   |    |   0 or more Applications
          | Application ID |    +
          +-----------------+
                 ^
                 |
                 |
                 v
              +-------------+
          +-------------+    |
          |I2RS Client  |    |
          |.............|    |   1 or more Clients
          |  Client ID  |    +
```

```
         +-------------+
                ^
                |
                |
                v
         +-------------+                 +-----------------------------+
         |I2RS Agent   |---------------->|Trace Log                    |
         |             |                 |.............................|
         +-------------+                 |Log Entry  [1 .. N]          |
               |  ^                      |.............................|
               |  |                      |Event ID                     |
               |  |                      |Starting Timestamp           |
               |  |                      |Request State                |
               |  |                      |Client ID                    |
               |  |                      |Client Priority              |
               |  |                      |Secondary ID                 |
   Operation + |  | Result Code          |Client Address               |
    Op Data    |  |                      |Requested Operation          |
               |  |                      |Applied Operation            |
               |  |                      |Operation Data Present       |
               |  |                      |Requested Operation Data     |
               |  |                      |Applied Operation Data       |
               |  |                      |Transaction ID               |
               |  |                      |Result Code                  |
               |  |                      |Ending Timestamp             |
               |  |                      |Timeout Occurred             |
               v  |                      |End Of Message               |
         +-------------+                 +-----------------------------+
         |Routing      |
         |System       |
         +-------------+
```

Figure 1: I2RS Interaction Trace Log Capture

5.2. I2RS Trace Log Fields

The following fields comprise an I2RS trace log. These fields ensure that each I2RS interaction can be properly traced back to the client that made the request at a specific point in time.

The list below describes the fields captured in the I2RS trace log. This list represents a common set of fields that MUST appear in all

I2RS trace logs. In addition to these fields, I2RS agent implementations MAY choose to log additional fields such as I2RS client vendor or agent statistics like free memory, performance metrics, etc.

Event ID: This is a unique identifier for each event in the I2RS trace log. An event can be a client authenticating with the agent, a client to agent operation, or a client disconnecting from an agent. Operation events can either be logged atomically upon completion (in which case they will have both a Starting and an Ending Timestamp field) or they can be logged at the beginning of each Request State transition. Since 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. If each state transition is logged for an operation, the same ID MUST be used for each of the Request State log entries. In this way, the life of a request can be easily followed in the I2RS trace log. Beyond the requirement that the Event ID MUST be unique for each event, the specific type and value is left up to the implementation.

Starting Timestamp: The specific time at which the I2RS operation enters the specified Request State within the agent. If the log entry covers the entire duration of the request, then this will be the time that it was first received by the agent. This field MUST be present in all entries that specify the beginning of the state transition, as well as those entries that log the entire duration of the request. The time is passed in the full timestamp format [RFC3339], including the date and offset from Coordinated Universal Time (UTC). Given that many I2RS operations can occur in rapid succession, the fractional seconds element of the timestamp MUST be used to provide adequate granularity. Fractional seconds SHOULD be expressed with at least three significant digits in second.microsecond format.

Request State: The state of the given operation within the I2RS agent state machine at the specified Starting or Ending Timestamps. The I2RS agent SHOULD generate a log entry at the moment a request enters and exits a state. Upon entering a new state, the log entry will have a Starting Timestamp set to the time of entry and no Ending Timestamp. Upon exiting a state, the log entry will have an Ending Timestamp set to the time of exit and no Starting Timestamp. The progression of the request through its various states can be linked using the Event ID. The states can be one of the following values:

PENDING: The request has been received and queued for processing.

IN PROCESS: The request is currently being handled by the I2RS agent.

COMPLETED: The request has reached a terminal point.

Every state transition SHOULD be logged unless doing so will put an undue performance burden on the I2RS agent. However, an entry with the Request State set to COMPLETED MUST be logged for all operations. If the COMPLETED state is the only entry for a given request, then it MUST have both Starting and Ending Timestamps that cover the entire duration of the request from ingress to the agent until completion.

Client Identity: The I2RS client identity used to authenticate the client to the I2RS agent.

Client Priority: The I2RS client priority assigned by the access control model that authenticates the client. For example, this can be set by the Network Configuration Protocol (NETCONF) Access Control Model (NACM) as described in [RFC6536].

Secondary Identity: This is an opaque identity that may be known to the client from a controlling network application. This is used to trace the network application driving the actions of the client. The client may not provide this identity to the agent if there is no external network application driving the client. However, this field MUST be logged even if the client does not provide a Secondary Identity. In that case, the field will be logged with an empty value.

Client Address: This is the network address of the client that connected to the agent. For example, this may be an IPv4 or an IPv6 address.

Requested Operation: This is the I2RS operation that was requested to be performed. For example, this may be an add route operation if a route is being inserted into a routing table. This may not be the operation that was actually applied to the agent.

In the case of a client authenticating to the agent, the Requested Operation MUST be "CLIENT AUTHENTICATE". In the case of a client disconnecting from the agent, the Requested Operation MUST be "CLIENT DISCONNECT".

Applied Operation: This is the I2RS operation that was actually performed. This can differ from the Requested Operation in cases where the agent cannot satisfy the Requested Operation. This field may not be logged unless the Request State is COMPLETED.

Operation Data Present: This is a Boolean field that indicates whether or not additional per-Operation Data is present.

Requested 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. If Operation Data is provided, then the Operation Data Present field MUST be set to TRUE. Some operations may not provide operation data. In those cases, the Operation Data Present field MUST be set to FALSE, and this field MUST be empty. This may not represent the data that was used for the operation that was actually applied on the agent.

When a client authenticates to the agent, the Requested Operation Data MUST contain the client priority. Other attributes such as credentials used for authentication MAY be logged.

Applied Operation Data: This field comprises the data that was actually applied as part of the Applied Operation. If the agent cannot satisfy the Requested Operation with the Requested Operation Data, then this field can differ from the Requested Operation Data. This field will be empty unless the Requested Operation Data was specified. This field may not be logged unless the Request State is COMPLETED.

Transaction ID: The Transaction Identity represents that this particular operation is part of a long-running I2RS transaction that can consist of multiple, related I2RS operations. Using this value, one can relate multiple log entries together as they are part of a single, overall I2RS operation. This is an optional field that may not be logged unless the event is part of a long-running transaction.

Result Code: This field holds the result of the operation once the Request State is COMPLETED. In the case of Routing Information Base (RIB) operations, this MUST be the return code as specified in Section 4 of [RIBINFO]. 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.

Timeout Occurred: This is a Boolean field that indicates whether or not a timeout occurred in the operation. When this is true, the value of the Ending Timestamp MUST be set to the time the agent recorded for the timeout occurrence. This field may not be logged unless the Request State is COMPLETED.

Ending Timestamp: The specific time at which the I2RS operation exits the specified Request State within the I2RS agent. If the log entry covers the entire duration of the request, then this will be the time that the request reached a terminal point within the agent. This field MUST be present in all entries that specify the ending of the state transition, as well as those entries that log the entire duration of the request. The time is passed in the full timestamp format [RFC3339], including the date and offset from Coordinated Universal Time (UTC). See the description for Starting Timestamp above for the proper format of the Ending Timestamp.

End Of Message: Each log entry SHOULD have an appropriate End Of Message (EOM) indicator. See 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-formatted log (similar to a syslog) the EOM marker may be a newline character. In an XML-formatted log, the schema would provide for element tags that denote the beginning and end of records. In a JSON-formatted log, the syntax would provide record separation (likely by comma-separated array elements).

6. Examples

This section shows a sample of what the fields and values could look like.

```
Event ID:                   1
Starting Timestamp:         2013-09-03T12:00:01.21+00:00
Request State:              COMPLETED
Client ID:                  5CEF1870-0326-11E2-A21F-0800200C9A66
Client Priority:            100
Secondary ID:               com.example.RoutingApp
Client Address:             2001:db8:c0c0::2
Requested Operation:        ROUTE_ADD
Applied Operation:          ROUTE_ADD
Operation Data Present:     TRUE
Requested Operation Data: PREFIX 2001:db8:feed:: PREFIX-LEN 64
                            NEXT-HOP 2001:db8:cafe::1
Applied Operation Data:     PREFIX 2001:db8:feed:: PREFIX-LEN 64
```

```
                                 NEXT-HOP 2001:db8:cafe::1
   Transaction ID:               2763461
   Result Code:                  SUCCESS(0)
   Timeout Occurred:             FALSE
   Ending Timestamp:             2013-09-03T12:00:01.23+00:00
```

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 operations that can be retrieved to troubleshoot I2RS-related impact to the network. Changes that occur to the network element's local configuration outside of the I2RS protocol that preempt I2RS state will only be logged if the network element notifies the I2RS agent.

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 operations 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. client requests may not always be processed synchronously or within a bounded time period. Consequently, to ensure that trace log fields, such as "Operation" and "Result Code", are part of the same trace log record, buffering of the trace log entries may be required. This buffering may result in additional resource load on the agent and the network element.

Section 7.3 discusses rotating the trace log in order to preserve the operation history without exhausting agent or network device resources. It is perfectly acceptable, therefore, to use both an in-memory buffer for recent operations while rotating or archiving older operations to a local file.

It is outside the scope of this document to specify the implementation details (i.e., size, throughput, data protection, etc.) for the physical storage of the I2RS log file. In terms of data retention, attention should be paid to the length of time that the I2RS trace log data is kept when that data contains security- or privacy-sensitive attributes. The longer this data is retained, the higher the impact if it were to be leaked. It is also possible that legislation may impose some additional requirements on the minimum and/or maximum durations for which some kinds of data may be retained.

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 are outside the scope of this document. However, it should be possible to do a file rotation based on either the 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 a vendor-agnostic and consistent interface to collect I2RS trace data. Correspondingly, retrieval of the data should also be made vendor-agnostic.

Despite the fact that export of I2RS trace log information could be an invaluable diagnostic tool for off-box analysis, exporting this information MUST NOT interfere with the ability of the agent to process new incoming operations.

The following three sections describe potential ways the trace log can be accessed. The use of I2RS pub/sub for accessing trace log data is mandatory-to-implement, while others are optional.

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. Syslog MAY be employed as a means of retrieving or disseminating the I2RS trace log contents.

If syslog is used for trace log retrieval, then existing logging infrastructure and capabilities of syslog [RFC5424] should be leveraged without the need to define or extend existing formats. That is, the various fields described in Section 5.2 SHOULD be modeled and encoded as Structured Data Elements (referred to as "SD-ELEMENT"), as described in Section 6.3.1 of [RFC5424].

7.4.2. Retrieval via I2RS Information Collection

Section 7.7 of the I2RS architecture [RFC7921] 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 7.6 of the I2RS architecture [RFC7921] goes on to describe notification mechanisms for a feed of changes happening within the I2RS layer. Specifically, the requirements for a publish-subscribe system for I2RS are defined in [RFC7923]. I2RS agents MUST support publishing I2RS trace log information to that feed as described in [RFC7923]. 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. The full range of potential activities is virtually limitless and the details of how they are performed are outside the scope of this document, however.

8. 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.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <http://www.rfc-editor.org/info/rfc3339>.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <http://www.rfc-editor.org/info/rfc5424>.

[RFC7921] Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", RFC 7921, DOI 10.17487/RFC7921, June 2016, <http://www.rfc-editor.org/info/rfc7921>.

[RFC7923] Voit, E., Clemm, A., and A. Gonzalez Prieto, "Requirements for Subscription to YANG Datastores", RFC 7923, DOI 10.17487/RFC7923, June 2016.

9.2. Informative References

[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>.

[RFC7920] Atlas, A., Ed., Nadeau, T., Ed., and D. Ward, "Problem Statement for the Interface to the Routing System", RFC 7923, DOI 10.17487/RFC7923, June 2016, <http://www.rfc-editor.org/info/rfc7920>.

[RIBINFO] Bahadur, N., Ed., Kini, S., Ed., and J. Medved, "Routing Information Base Info Model", Work in Progress, draft-ietf-i2rs-rib-info-model-08, October 2015.

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, Dean Bogdanovich, Ignas Bagdonas, Nobo Akiya, Kwang-koog Lee, Sue Hares, Mach Chen, Alex Clemm, Stephen Farrell, Benoit Claise, Les Ginsberg, Suresh Krishnan, and Elwyn Davies for their reviews, contributed text, and suggested improvements to this document.

Authors' Addresses

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

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

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

Email: gsalguei@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com