

Workgroup: RIFT
Internet-Draft: draft-ietf-rift-kv-registry-07
Published: 20 October 2023
Intended Status: Standards Track
Expires: 22 April 2024
Authors: J. Head, Ed. T. Przygienda
 Juniper Networks Juniper Networks
 RIFT Key/Value Structure and Registry

Abstract

The RIFT (Routing in Fat-Trees) protocol allows for key/value pairs to be advertised within Key-Value Topology Information Elements (KV-TIEs). The data contained within these KV-TIEs can be used for any imaginable purpose. This document defines the various Key-Types (i.e. Well-Known, OUI, and Experimental) and a method to structure corresponding values.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

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 <https://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 22 April 2024.

Copyright Notice

Copyright (c) 2023 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

- [1. Introduction](#)
- [2. Key Structure](#)
 - [2.1. Experimental Key-Type](#)
 - [2.2. Well-Known Key-Type](#)
 - [2.3. OUI Key-Type](#)
- [3. Design Considerations](#)
 - [3.1. Tie-Breaking Considerations](#)
 - [3.1.1. Southbound Key-Value TIE Tie-Breaking Key/Value Pair](#)
 - [3.2. Key Target](#)
 - [3.2.1. Key Target Processing](#)
- [4. IANA Considerations](#)
 - [4.1. RIFT Key-Types](#)
 - [4.1.1. RIFT Key-Types Requested Entries](#)
 - [4.2. RIFT Well-Known Key-Types](#)
 - [4.2.1. RIFT Well-Known Key-Types Requested Entries](#)
 - [4.3. Expert Review Guidance](#)
- [5. Security Considerations](#)
- [6. Acknowledgements](#)
- [7. Normative References](#)
- [8. Informative References](#)
- [Appendix A. Thrift Models](#)
 - [A.1. southbound_kv.thrift](#)
- [Authors' Addresses](#)

1. Introduction

The Routing in Fat-Trees [RIFT](#) [[RIFT](#)] protocol allows for key/value pairs to be advertised within Key-Value Topology Information Elements (KV-TIEs). There are no restrictions placed on the type of data that is contained in KV-TIEs nor what the data is used for.

For example, it might be beneficial to advertise overlay protocol state from leaf nodes to the Top-of-Fabric (ToF) nodes. This would make it possible to view critical state of a fabric-wide service from a single ToF node rather than retrieving and reconciling the same state from multiple leaf nodes.

2. Key Structure

This section describes the generic Key structure and semantics, [Figure 1](#) further illustrates these components.

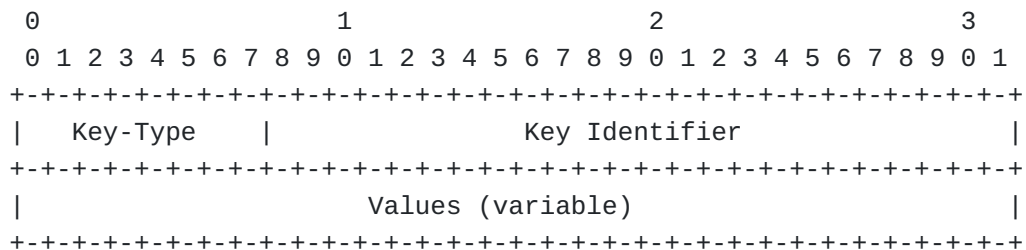


Figure 1: Generic Key-Value Structure

where:

Key-Type:

A 1-byte value that identifies the Key-Type. It MUST be a reserved value from the RIFT Key-Type Registry that is defined later in this document.

The range of valid values is 1 - 255 (2^8-1).

0 is an illegal value and MUST NOT be allocated to or used by any implementation. It MUST be ignored on receipt.

Key Identifier:

A 3-byte value that identifies the specific key and describes the structure of the contained values.

The range of valid values is 1 - 16777215 ($2^{24}-1$).

0 is an illegal value and MUST NOT be allocated to or used by any implementation. It MUST be ignored on receipt.

Values:

A variable length value that contains data associated with the Key Identifier. It SHOULD contain 1 or more elements. Whether the collection of elements allows duplicates and/or is ordered is governed by the particular Key Identifier's specification.

2.1. Experimental Key-Type

This section reserves a value in the RIFT Key-Type Registry to indicate an Experimental Key-Type.

As shown in [Figure 2](#), the Key-Type will be used to identify the Key-Type as Experimental. The Key Identifier will be used to identify the specific key and describe the structure of the contained values.

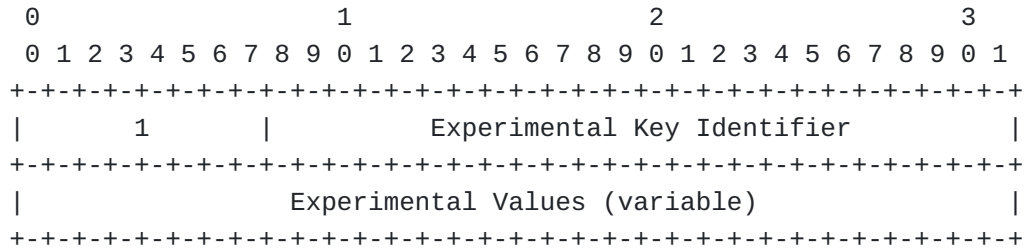


Figure 2: Experimental Key-Type

2.2. Well-Known Key-Type

This section reserves a value in the RIFT Key-Type Registry to indicate Well-Known Key-Types that all implementations SHOULD support.

As shown in [Figure 3](#), the Key-Type will be used to identify the Key-Type as Well-Known. The Key Identifier will be used to identify the specific key and describe the structure of the contained values.

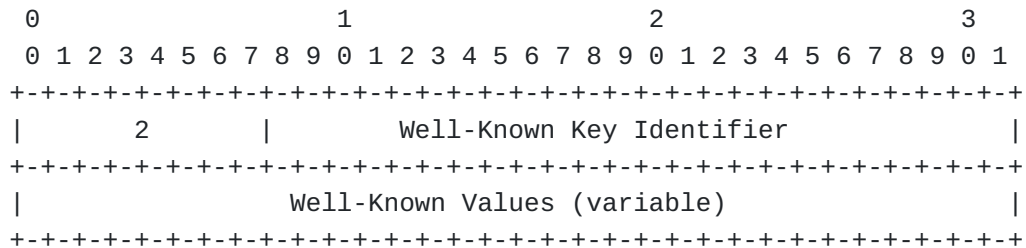


Figure 3: Well-Known Key-Type

2.3. OUI Key-Type

This section reserves a value in the RIFT Key-Type Registry to indicate an OUI (vendor-specific) Key-Type that any implementation MAY support.

As shown in [Figure 4](#), the Key-Type will be used to identify the Key-Type as OUI. The Key Identifier MUST use the implementing organization's reserved OUI space to indicate the key and value structure.

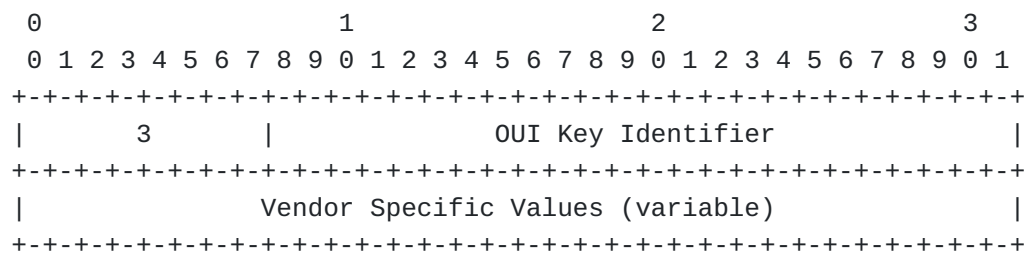


Figure 4: OUI Key-Type

3. Design Considerations

While no restrictions are placed on Key-Value data or what it is used for, it is RECOMMENDED that a serialized [Thrift](#) [[THRIFT](#)] model be used for simpler interoperability. [[RIFT-AUTO-EVPN](#)] is an example of this type of implementation.

Key-Value elements SHOULD NOT be used to carry topology information used by RIFT itself to perform distributed computations.

3.1. Tie-Breaking Considerations

In cases where KV-TIEs are flooded from north to south, policies SHOULD be implemented in order to avoid network-wide flooding.

For networks with more than one ToF node, it is RECOMMENDED that those ToF nodes contain identical KV-TIE information when being distributed from north to south. [RIFT](#) [[RIFT](#)] requires that only one KV-TIE is selected when identical keys are received from multiple northbound neighbors. If this is not considered then the tie-breaking rules may cause a node to select a suboptimal KV-TIE. Consider a case where failure conditions cause the ToF nodes to become split-brained. While the Key-Type and Key Identifier will be identical, the value(s) contained within may differ. The node(s) receiving these differing KV-TIEs will select the one from the ToF node with the highest System ID, potentially leading to unintended effects.

3.1.1. Southbound Key-Value TIE Tie-Breaking Key/Value Pair

This Key/Value pair contains information that allows for verification of proper tie-breaking for the Southbound Key store.

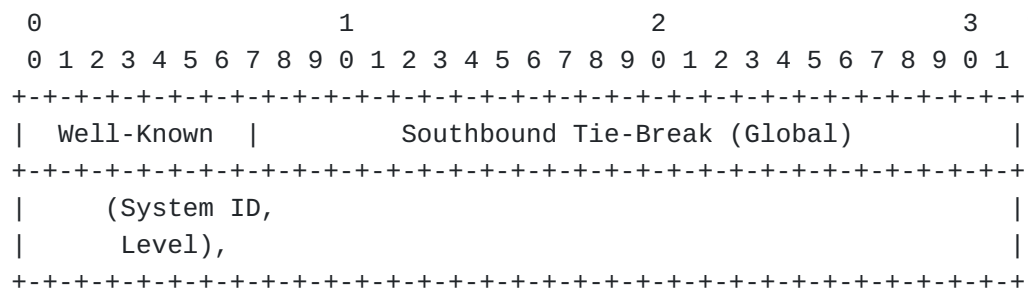


Figure 5: Southbound Tie-Break (Global) Key/Value Pair

where:

System ID:

A REQUIRED value indicating the node's unique System ID.

Level:

A RECOMMENDED value indicating the node's level.

3.2. Key Target

The Key Target is an optional value that identifies group(s) of node(s) that are intended to receive a given Key-Value TIE. Key Targets are 64-bits in length with a valid range of 0 - 18446744073709551615 ($2^{64}-1$), this will reduce the chances that Key Target values collide.

A value of all 0s represent that every node is intended to receive this Key-Value TIE and MUST NOT be used for any other reason.

A value of all 1s represent that all leaf nodes are intended to receive this Key-Value TIE and MUST NOT be used for any other reason.

Any other value MUST be derived from the following normative algorithm.

```

/// random seeds used in algorithms to increase entropy
pub const RANDOMSEEDS: [UnsignedSystemID; 3] = [
    67438371571u64,
    37087353685,
    88675895388,
];

/// given a system ID delivers the bits set by the according Bloom Filter
/// key value target.
///
/// @note: This is standardized and cannot be changed between releases!
pub (crate) fn target2bits(target: UnsignedSystemID) -> KeyValueTargetType
(0 as usize .. 3)
    .map(|s| {
        let rot = (target ^ RANDOMSEEDS[s]).rotate_left(s as _);
        rot.to_ne_bytes().iter().fold(0, |v: u8, nv| v.rotate_right(
        })
        .fold(0, |v, nv| v | v | (1 << nv))
    }
}

```

Figure 6: Key Target Standard Algorithm

3.2.1. Key Target Processing

Nodes that support the processing of Key Targets MUST only do so on KV-TIEs in the southbound direction. Key Targets MUST NOT be present on KV-TIEs in the northbound direction and are otherwise ignored and logged.

Nodes that do not support the processing of Key Targets MUST continue to send KV-TIEs to all nodes in the appropriate direction. Additionally, Key Targets MUST be preserved when KV-TIEs are re-originated in the southbound direction.

3.2.1.1. Purging/Rollover

There are several reasons a node may select a different KV-TIE. For example, the KV-TIE is considered newer due to the sequence number incrementing, there was a change in the original tie-breaking result between multiple KV-TIEs, or a loss of northbound connectivity to the node that advertised the previously selected KV-TIE.

Consider a case where Leaf-1, Leaf-2, and Leaf-3 are members of a group of nodes represented by Key Target KT1. If Leaf-2 is removed from that group and a newer instance of the KV-TIE needs to be flooded Leaf-2 will have to maintain the older KV-TIE in the LSDB until the lifetime expires. This could lead to suboptimal behavior in the fabric.

If the new KV-TIE being flooded does not include the previous Key Target value, then implementations SHOULD flood the newer instance of the KV-TIE with a very short lifetime to nodes that belonged to the previous Key Target but not the new Key Target. This logic only applies to KV-TIEs being flooded in the southbound direction.

4. IANA Considerations

Per [RFC8126], IANA is requested to create two new registries under the top-level "RIFT" category:

*RIFT Key-Types

*RIFT Well-Known Key-Types

The following sections detail each registry's individual requirements and suggested values.

Experts reviewing requests for new values to either registry MUST consider the items in the [Expert Review Guidance](#) ([Section 4.3](#)) section.

4.1. RIFT Key-Types

This section requests that IANA create and help govern the following registry:

Registry Name:

RIFT Key-Types

Registration Procedures:

Expert Review

Description:

Key-Type registry for the RIFT protocol.

Reference:

This document.

4.1.1. RIFT Key-Types Requested Entries

This section requests that IANA register the following suggested values to the "RIFT Key-Types" registry.

Value	Key-Type	Description	Status/ Reference
0	Illegal	Not allowed.	This document
1	Experimental	Indicates that the Key-Type is Experimental.	This document.

Value	Key-Type	Description	Status/ Reference
2	Well-Known	Indicates that the Key-Type is Well-Known.	This document.
3	OUI	Indicates that the Key-Type is OUI (vendor specific).	This document.

Table 1

4.2. RIFT Well-Known Key-Types

This section requests that IANA create and help govern the following registry:

Registry Name:

RIFT Well-Known Key-Types

Registration Procedures:

Expert Review

Description:

Well-Known Key-Types registry for the RIFT protocol.

Reference:

This document.

4.2.1. RIFT Well-Known Key-Types Requested Entries

This section requests that IANA register the following suggested values to the "RIFT Well-Known Key-Types" Registry.

Value	Key-Identifier	Description	Status/ Reference
0	Illegal	Not allowed.	This document.
1	MAC/IP Binding	To be defined.	To be defined.
2	FAM Security Roll-Over Key	To be defined.	To be defined.
127	Southbound Tie-Break Key	Used for Southbound Keystore tie-breaking purposes.	This document.

Table 2

4.3. Expert Review Guidance

Experts reviewing requests for values from the "RIFT Key-Types" registry or the "RIFT Well-Known Key-Types" registry are responsible for the following:

1. Determining the existence of a specification that clearly defines the purpose supporting the request and MUST contain all required fields for given registry.

The document MUST also be permanent and publically available.

2. Ensuring that any requests are made available to the RIFT working group for review should the work originate from outside of the RIFT Working Group.
3. Ensuring that any work produce outside of the IETF does not conflict with any work that is already published or actively pursuing being published.

5. Security Considerations

This document introduces no new security concerns to RIFT or other specifications referenced in this document given that the Key-Value TIEs are already extensively secured by the [RIFT](#) [[RIFT](#)] protocol specification itself.

6. Acknowledgements

To be provided.

7. 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RIFT] Przygienda, T., Sharma, A., Thubert, P., Rijsman, B., Afanasiev, D., and J. Head, "RIFT: Routing in Fat Trees", Work in Progress draft-ietf-rift-rift-19, October 2023,

<<https://www.ietf.org/archive/id/draft-ietf-rift-rift-19.html>>.

8. Informative References

[**RIFT-AUTO-EVPN**] Head, J., Przygienda, T., and W. Lin, "RIFT Auto-EVPN", Work in Progress, draft-ietf-rift-auto-evpn-03, December 2022, <<https://www.ietf.org/archive/id/draft-ietf-rift-auto-evpn-03.html>>.

[**THRIFT**] Apache Software Foundation, "Thrift Language Implementation and Documentation", <<https://github.com/apache/thrift/tree/0.15.0/doc>>.

Appendix A. Thrift Models

This section contains the Thrift models that MAY be used to test southbound Key-Value tie-breaking based on System ID. Per the main [RIFT](#) [[RIFT](#)] specification, all signed values MUST be interpreted as unsigned values.

A.1. southbound_kv.thrift

```
include "common.thrift"

namespace py southbound_kv
namespace rs models

const i8 GlobalSystemIdentifierKV = 127

/** simple type to test correct tie-breaking based on system ID */
struct SystemIdentifierKV {
    1: required common.SystemIDType system_id,
    2: optional common.LevelType level,
}
```

Figure 7: RIFT Common Schema for Southbound Key-Value Tie-Break Key-Type

Authors' Addresses

Jordan Head (editor)
Juniper Networks
1137 Innovation Way
Sunnyvale, CA
United States of America

Email: jhead@juniper.net

Tony Przygienda
Juniper Networks
1137 Innovation Way
Sunnyvale, CA
United States of America

Email: prz@juniper.net