Authors: J. Head, Ed.   T. Przygienda
         Juniper Networks   Juniper Networks

RIFT Keys Structure and Well-Known Registry in Key Value TIE

Abstract

Routing in Fat-Trees RIFT [RIFT] 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", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [RFC2119].

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 24 March 2022.

Copyright Notice

Table of Contents

1. Description

Routing in Fat-Trees (RIFT [RIFT]) 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.

This document defines a Key Type Registry to maintain Well-Known and vendor specific Key Types in order to simplify interoperability between implementations and eliminate the risk of collision for future implementations. An Experimental Key Type is additionally defined.

2. Key-Value Pair Structure

Figure 1 illustrates the generic Key-Value Pair structure.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Key-Type    |              Key Identifier                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Values (variable)                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

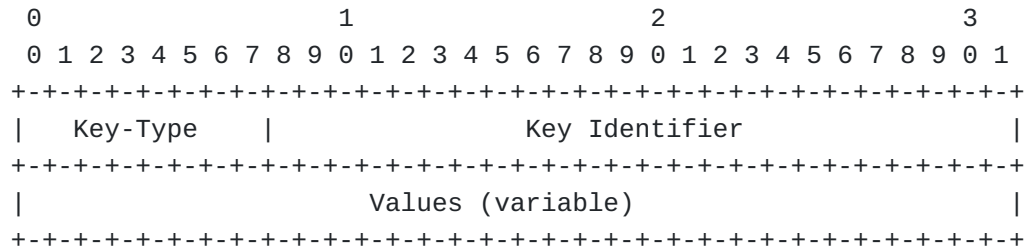   Figure 1: Generic Key-Value Structure

   where:

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

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

      Values:
         A variable length value that contains data associated with the
         Key. 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.

2.1. Well-Known Key Type

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

   As shown in Figure 2, the Key-Type will be used to identify that the
   Key Type is Well-Known. The Key Identifier will be used to identify
   the specific Key and describe the structure of the contained values.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     TBD2      |         Well-Known Key Identifier             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Well-Known Values (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
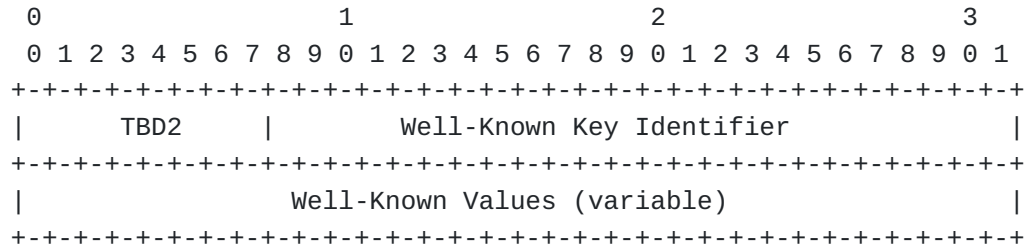
                      Figure 2: Well-Known Key Type

2.2. OUI Key Type

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

   As shown in Figure 3, the Key-Type will be used to identify the Key
   Type as OUI. The Key Identifier MUST use an organization's reserved
   OUI space to indicate the Key and value structure.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     TBD3      |       Organizationally Unique Identifier      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Vendor Specific Values (variable)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
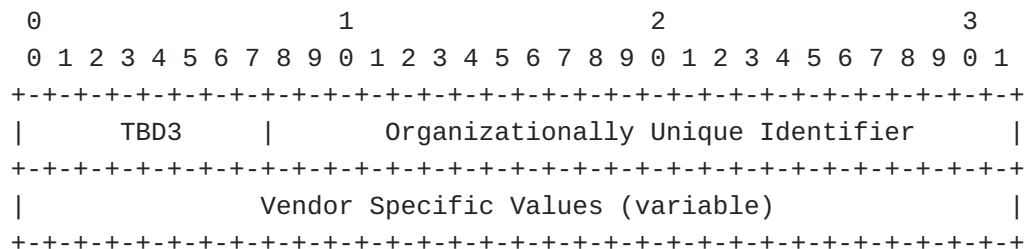
                       Figure 3: OUI Key Type

2.3. Experimental Key Type

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

   As shown in Figure 4, the Key-Type will be used to identify the Key
   Type as Experimental. The Key Identifier will be used to identify
   the specific experimental Key and describe the structure of the
   contained values.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     TBD1      |          Experimental Key Identifier          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Experimental Values (variable)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
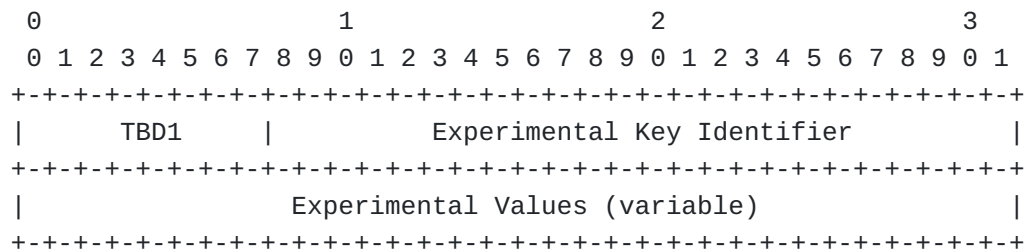
                   Figure 4: Experimental Key Type

3. IANA Considerations

   This section requests that IANA help govern Key Types via the usual
   IANA registry procedures as per [RFC8126].

All values not suggested are available for assignment. The
allocation of new values MUST be done via "Expert Review"
procedures.

3.1. Key Type Registry

This section defines the Key Type Registry that is used to identify
a specific Key Type. It also suggests values for Experimental, Well-
Known, and OUI Key Types.

The range of valid values is 1 - 255.

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

3.1.1. Requested Entries

Key Type	Value	Description
Experimental	TBD1	Indicates that the Key is Experimental.
Well-Known	TBD2	Indicates that the Key is Well-Known.
OUI	TBD3	Indicates that the Key is OUI.

Table 1

3.2. Experimental Key Type

This value indicates that a specific key is Experimental.

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 reception.

3.2.1. Requested Entries

Experimental Key	Identifier	Description
Illegal	0	Not allowed.

Table 2

3.3. Well-Known Key Type

This value indicates that a specific key is Well-Known.

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 reception.

3.3.1. Requested Entries

Well-Known Key	Identifier	Description
Illegal	0	Not allowed.
MAC/IP Binding	TBD1	To be defined.
FAM Security Roll-Over Key	TBD2	To be defined.

<div align="center">Table 3</div>

3.4. OUI Key Type

This value indicates a specific OUI Key using an organization's
reserved OUI space.

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 reception.

3.4.1. Requested Entries

OUI Key	Identifier	Description
Illegal	0	Not allowed.

<div align="center">Table 4</div>

4. Operational Considerations

While no restrictions are placed on Key-Value data or what it is
used for, it is RECOMMENDED that a serialized Thrift model be used
for simpler interoperability. RIFT Auto-EVPN [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.

In cases where Key-Value 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 Key-Value TIE information when
being distributed from north to south as the Key-Value tie breaking
rules in RIFT [RIFT] ultimately mention that only one Key-Value TIE
can be selected from multiple northbound neighbors. If this is not
considered, nodes receiving varying Key-Value TIEs might select a
suboptimal Key-Value TIE.

5. Security Considerations

This document introduces no new security concerns to RIFT or other
specifications referenced in this document given that the TIEs that

carry KV pairs are already extensively secured by the RIFT [RIFT] 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>.

[RIFT]   Przygienda, T., Sharma, A., Thubert, P., Rijsman, B.,
            and D. Afanasiev, "RIFT: Routing in Fat Trees", Work in
            Progress, draft-ietf-rift-rift-13, July 2021.

[RIFT-AUTO-EVPN] Head, J., Przygienda, T., and W. Lin, "RIFT Auto-
            EVPN", Work in Progress, draft-head-rift-auto-evpn-01,
            July 2021.

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