RIFT Key/Value TIE Structure and Processing

draft-ietf-rift-kv-tie-structure-and-processing-01

IETF 120

Vancouver

Jordan Head
(presenting)
Tony Przygienda

Non-

What’s new in version 01?

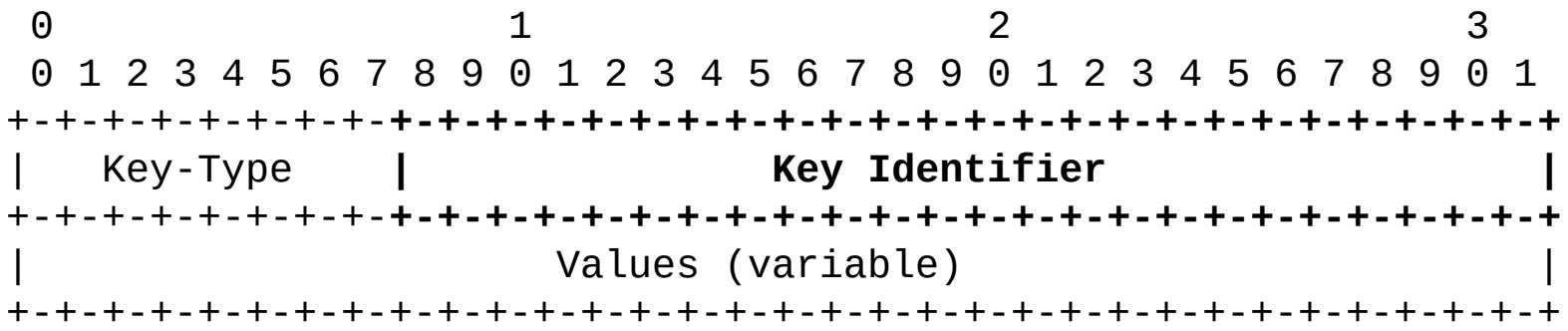
- **Introduction of Key Sub-Type**
- **Normative Language Fixes**
- **References**

What’s the Key Sub-Type?

- Discussions with Jeffrey revealed a new problem with respect to how we use the Key Identifier.
- Currently, the Key Identifier is somewhat analogous to a TLV.
- This alone will not provide an optimal way to distinguish between similarly structured KV-TIEs that carry different information.
  - For example, N-KV-TIEs will maintain originator’s System ID, but what if we need more than “node-level” specificity?
- Thus, we introduce the Key Sub-Type, which can be thought of as a Sub-TLV.

What does the Key Sub-Type look like?

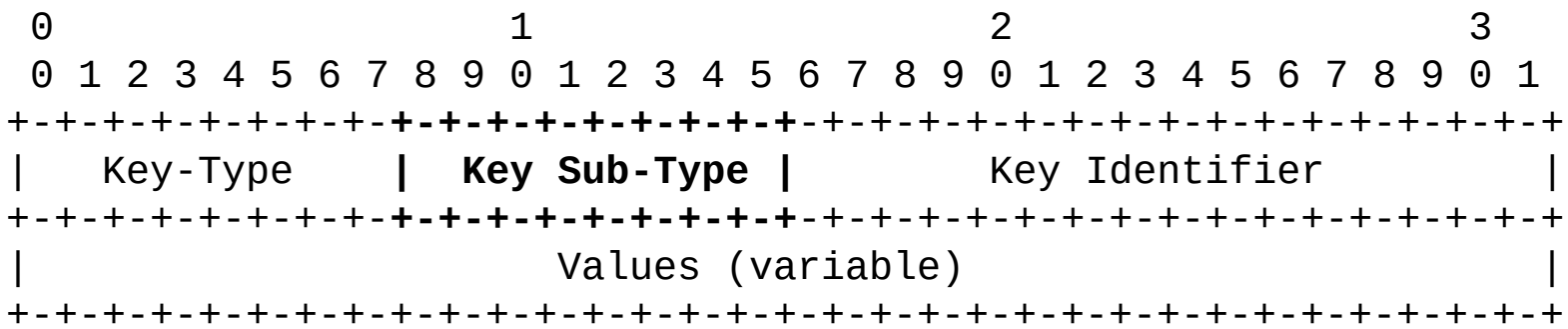
General Key-Value TIE Structure



- 3-bytes for the Key Identifier

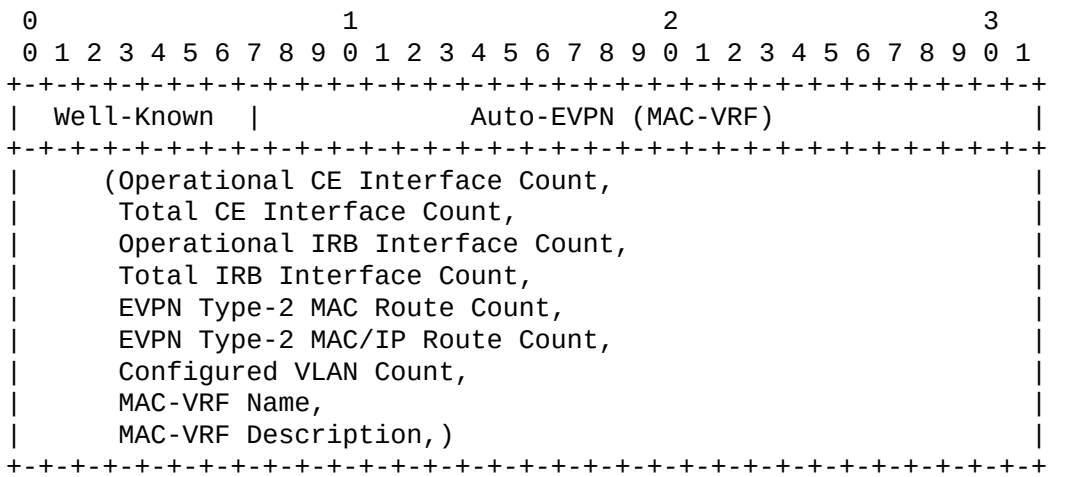
What does the Key Sub-Type look like?

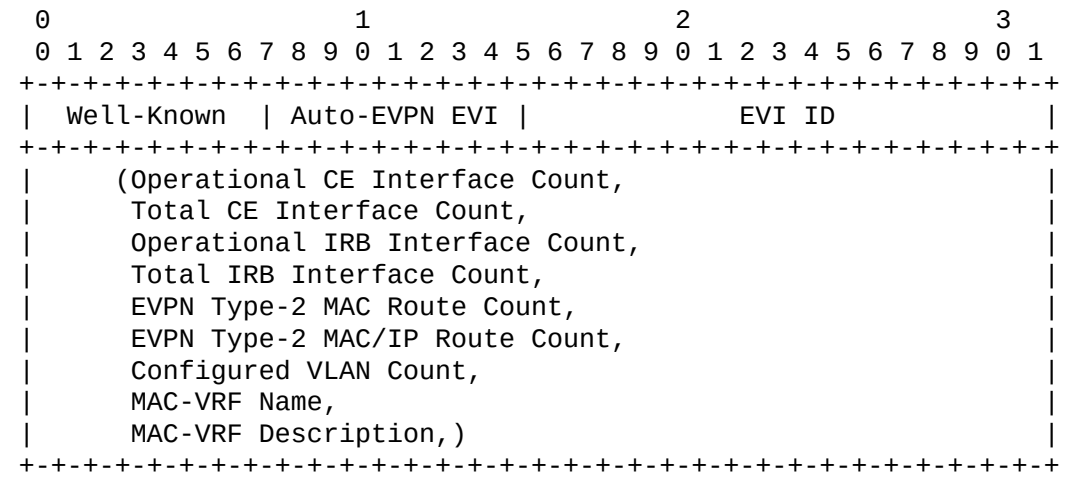
Key-Value TIE Structure with Key Sub-Type



- For Well-Known and Experimental Key-Types, we change the Key Identifier structure:
  - 1-byte for the Key Sub-Type
  - 2-bytes for the Key Identifier
  - The Key Identifier for the OUI Key-Type will remain at 3-bytes
- Key Sub-Type is OPTIONAL, but MUST be used for Well-Known and Experimental Key-Types
- New Key-Types can still be reserved for applications that may require 3-bytes of Key Identifier

In reality, we're already using Key Sub-Type





- Previously, Auto-EVPN (-05, pictured left) used all 3-bytes of Key Identifier to say that the KV-TIE contained EVI specific information from a particular node.
  - Additional language in the draft specified that the EVI ID MUST be encoded in the last 2 bytes of the Key Identifier

- Now, Auto-EVPN (-06, pictured right) *properly* reflects the desired structure that leverages the newly specified Key Sub-Type.

What else changed?

- **Normative Fixes**
  - The previous version stated that the Key Target was optional, rather than OPTIONAL.
- **Southbound TIE-Break Key/Value Pair**
  - Clarified that it is used to test an implementations tie-breaking behavior.
  - Adjusted structure to use Key Sub-Type.
- **References, etc.**

Questions?

Non-