Encoding Forwarding Actions in MPLS Labels Using a Multi-Purpose Special Purpose Label

Kireeti Kompella | Tarek Saad | Vishnu Pavan Beeram | Israel Meilik

Colby Barth | Chandra Ramachandran | Srihari Sangli |*Swamy SRK*

JUNIPER NETWORKS | Engineering Simplicity

Motivation

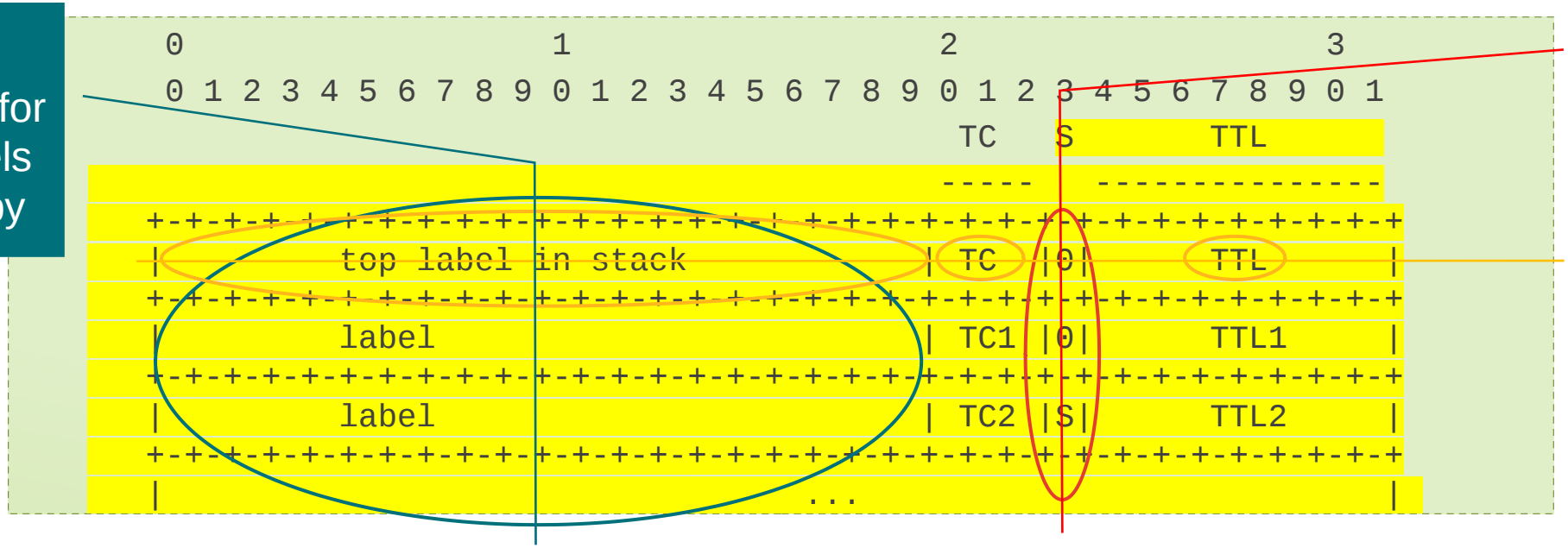
- There are multiple MPLS based features that require one or more actions to be applied in the forwarding plane along the path of an LSP
- Each of these features requires specific data to be carried in the MPLS packet
  - Some suffice in themselves, with no additional data
    - E.g., NFFRR – effectively, a boolean
  - Some features require further data to be carried
    - Which Slice-Aggregate does this packet belong to? (slide identifier)
    - Which SR path/Flow does this packet belong to? (path/flow ID)
    - Are there Generic Delivery Functions? (frag, etc.)
    - Is there OAM data within or beyond the MPLS label stack? (multiple types)
  - In the future, there may be other forwarding actions to consider
    - “Opaque” data offers a way to capture these
- Each of these would like one or more bSPLs as their “indicator”

© 2021 Juniper Networks

Juniper Business Use Only

Key Insight: Behavior of Forwarding Engines on Label Stack

Process label values (20 bits) for accessible labels for SPL, entropy



MUST process EoS bits

Process entire Top of Stack label

- For labels below the top of stack, forwarding engines look only at the label values (to compute a hash, or to look for special purpose labels) and the S bit (to find end of stack)
- They *don't* look at the TC bits or the TTL value of labels below top of stack
- In which case, we can repurpose these bits
- We MUST ensure that such labels never reach the top of stack (by popping labels above)

© 2021 Juniper Networks

Juniper Business Use Only

Approach

- ~~Defining a new special-purpose / extended-special-purpose label for every new feature is not ideal~~
  - This is expensive, generally involving a hardware re-spin or microcode update
- Proposal: use a single bSPL to *compactly* encode *multiple* forwarding actions in MPLS labels
  - This would serve as bSPL for EL, GISS, NFFRR, OAM, Path Identifier and Generic Delivery Function
  - EL already has an ELI, but the *rest are active requests* (except for GDFH, which may join in)
  - This bSPL (called FAI) uses the TC/TTL bits to encode more data
  - FAI is accompanied by forwarding actions data (FAD), also compactly encoded
  - This approach can serve as the blueprint for future bSPLs, again using the TC/TTL fields to encode more information
  - There is a major caveat: a bSPL encoded thus ***cannot*** appear as the top label in the stack
- A key new use case is the Global Identifier for Slice Selector (GISS)
  - Enables MPLS-based transport network slices (cf. draft-bestbar-teas-ns-packet)

© 2021 Juniper Networks

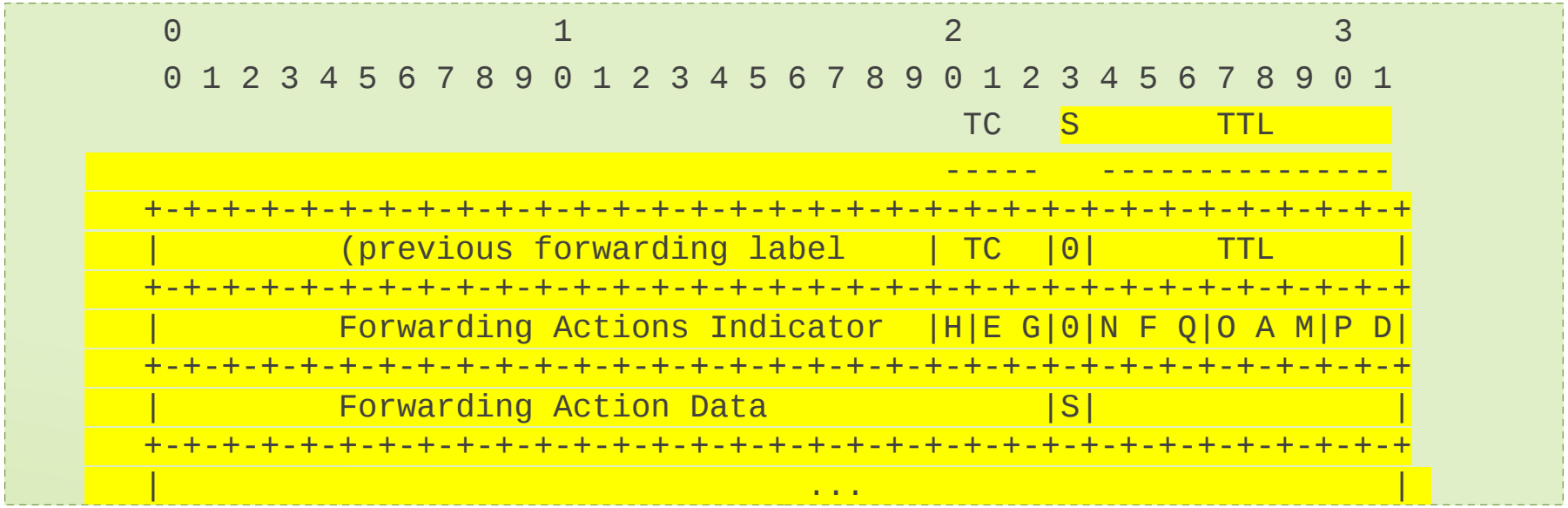
Juniper Business Use Only

Solution: FAI/FAD

- Use a bSPL called "Forwarding Actions Indicator” (FAI, value TBD) to signal that certain forwarding actions must be taken, and provide data for them (FAD)
- The data is divided into those that are part of the label stack (LS FAD), and those that come after the label stack (in the payload) (PL FAD)
  - Each section has a well-known part and an opaque part
  - Ignore the “opaque” part for now
- If PHP is done, the PHP router MUST push an Explicit NULL (label value 0 or 2) on the stack; the FAI MUST NOT show up at the top of the label stack
  - There MUST NOT be a TC set or TTL copy to the FAI from previous (popped) label
- If FAI shows up at the top of stack, the packet gets dropped

© 2021 Juniper Networks
Juniper Business Use Only

Forwarding Actions Ind (FAI) / Fwding Actions Data (FAD)

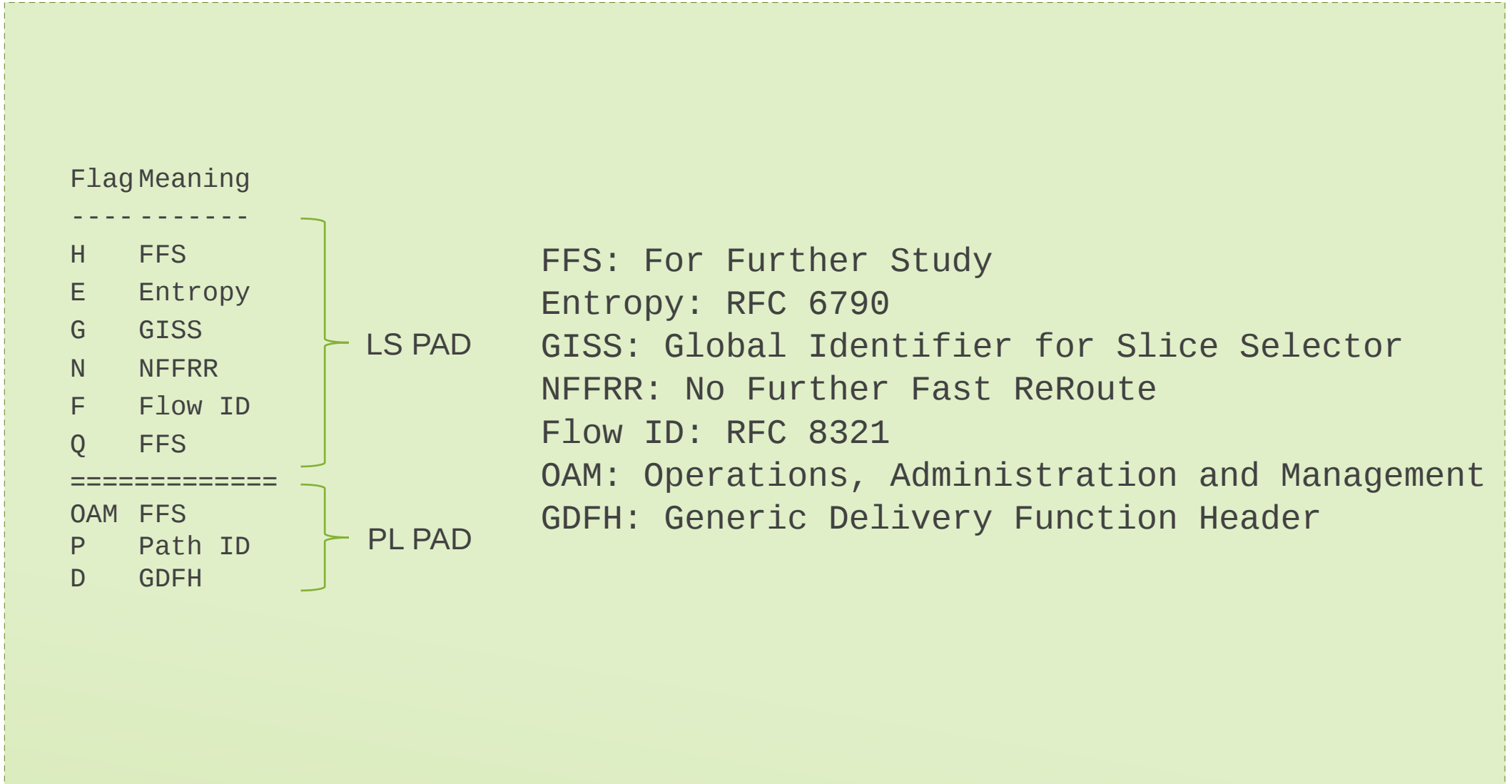


- Forwarding Actions Indicator: This is an MPLS base Special Purpose Label (bSPL). It indicates the presence of LS Forwarding Actions Data (FAD) immediately below this label stack entry in the label stack.
  - The TC and TTL fields of the FAI encode the types of FAD
  - The “end of stack” bit is sacrosanct and must reflect the end of label stack correctly
- Forwarding Actions Data: This includes “well-known” data and opaque data

© 2021 Juniper Networks

Juniper Business Use Only

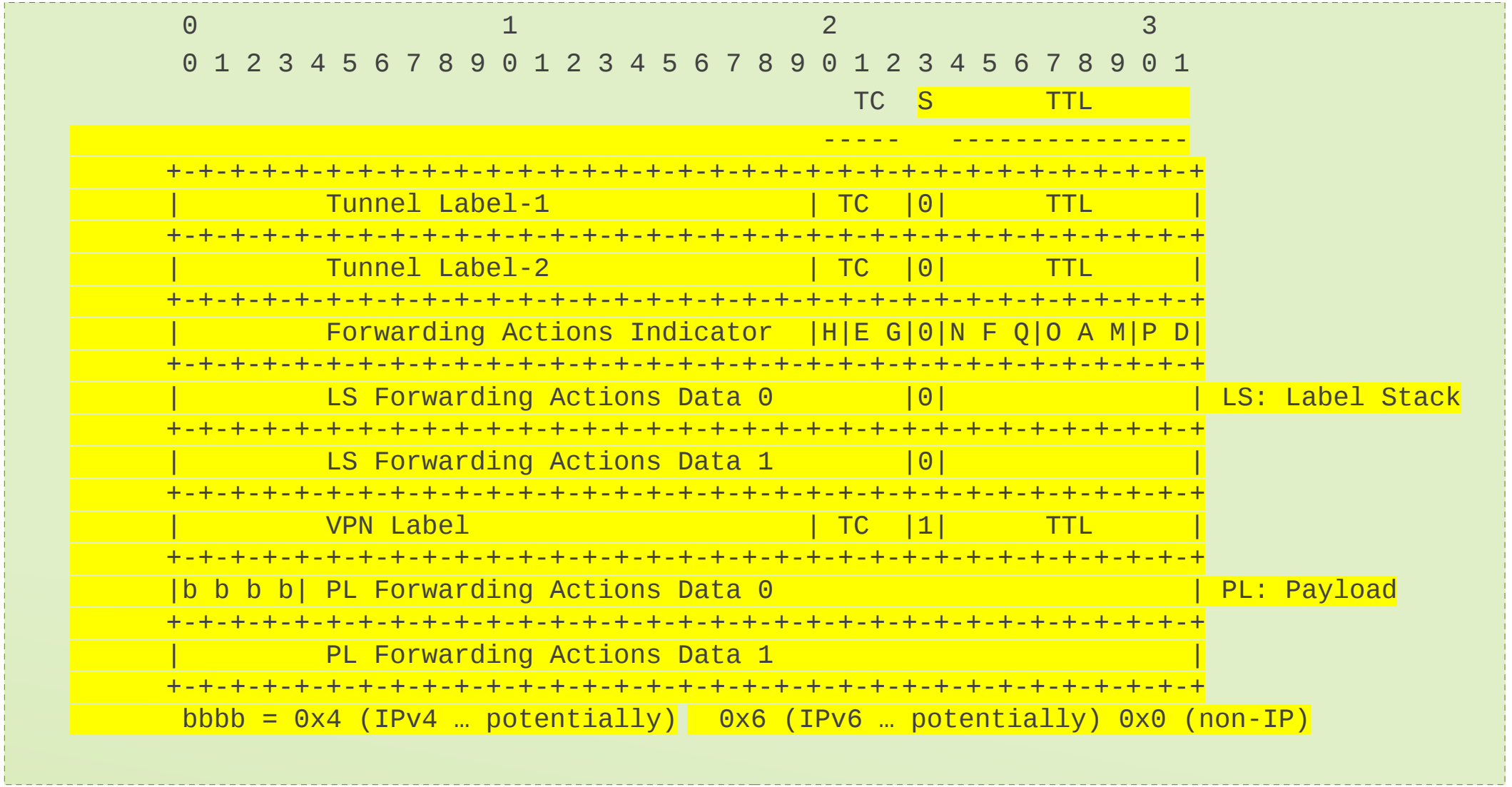
Example: FAI/FAD in Label Stack (LS) & in Payload (PL)



© 2021 Juniper Networks

Juniper Business Use Only

Example: FAI/FAD in Label Stack (LS) & in Payload (PL)



© 2021 Juniper Networks

Juniper Business Use Only

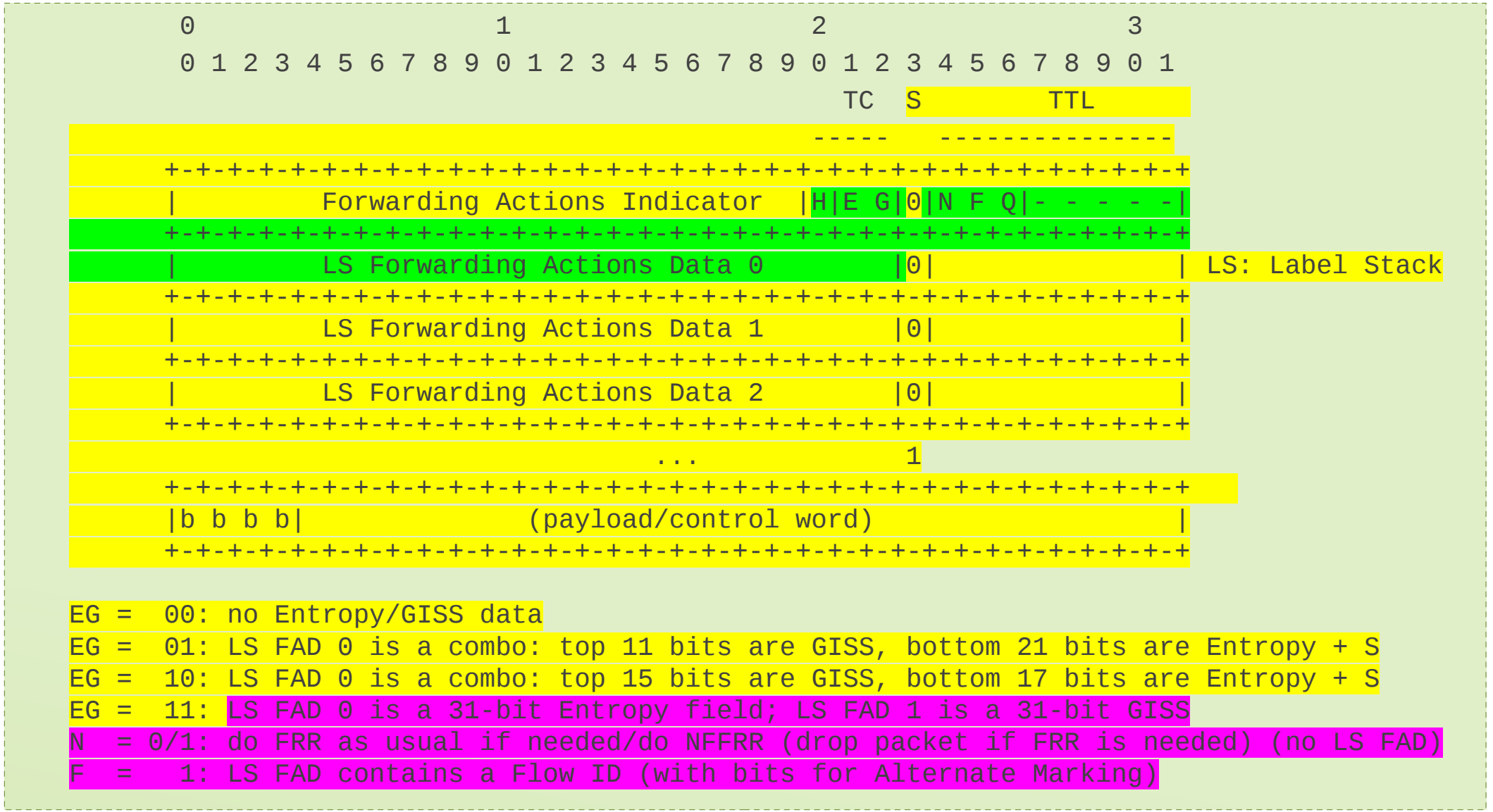
Notes

- The H bit is for an extended Forwarding Actions Header (FAH) – more flags, etc. (TBD)
- The contents of the LS FAD is determined by the LS FAD flag bits EGNQ
  - Ignore the Q flag (Opaque LS FAD) for now
  - These flags indicate the presence and order of items in the LS FAD
- The R bit is reserved for future use
- The contents of the PL FAD is determined by the PL FAD flag bits OAMPDq
  - Ignore q (Opaque PL FAD) for now
  - These flags indicate the presence and order of items in the PL FAD
- The first nibble of the payload MUST NOT be 0x4 or 0x6 if used for PL FAD
  - Hardware in the field may use this for load balancing info
  - If the first field in the payload is iOAM, this nibble is 0x1

© 2021 Juniper Networks

Juniper Business Use Only

How the LS FAD Flag Bits Are Used



© 2021 Juniper Networks

Juniper Business Use Only

EG Field

Entropy is used for load balancing (from RFC 6790)

GISS = Global Identifier for Slice Selector: data field used for QoS classification and path steering

A GISS value of 0 means “default slice” – use the TC bits for QoS

If the EG field is 00, the FAD carries neither an Entropy field nor a GISS

If the EG field is 11, the FAD carries both an Entropy field and a GISS, each 31 bits long

- This means that FAD is at least 2 words

The other values are a combo, where a single word contains both GISS and Entropy, with different splits

If you want NFFRR and Entropy and no GISS: set N to 1 and EG to 01 or 10 (20 or 16 bits of entropy)

- set the GISS to 0 (default)
- FAI + FAD = 2 words (can replace ELI + EL and (eventually) deprecate ELI)

© 2021 Juniper Networks

Juniper Business Use Only

Added Depth to Label Stack

Desired Actions	FAI + length of LS FAD	Dedicated bSPL
NFFRR	1	1
EL	2	2
NFFRR + EL	2	3
NFFRR + EL + GISS	2/3	6
NFFRR + FI	2	3
FI	2	2
FI + EL	3	4

© 2021 Juniper Networks

Juniper Business Use Only

Repeating the FAI/FAD for “Readable Label Depth”

To accommodate routers that have a limited “lookahead” for reading labels, the FAI can be repeated

There are a few alternatives here: the repeated FAIs can set EG to 00, so it’s a single word

- The “last” FAI has these bits set correctly, with the correct GISS and Entropy
- Other routers can search for the last FAI to get the real GISS and Entropy

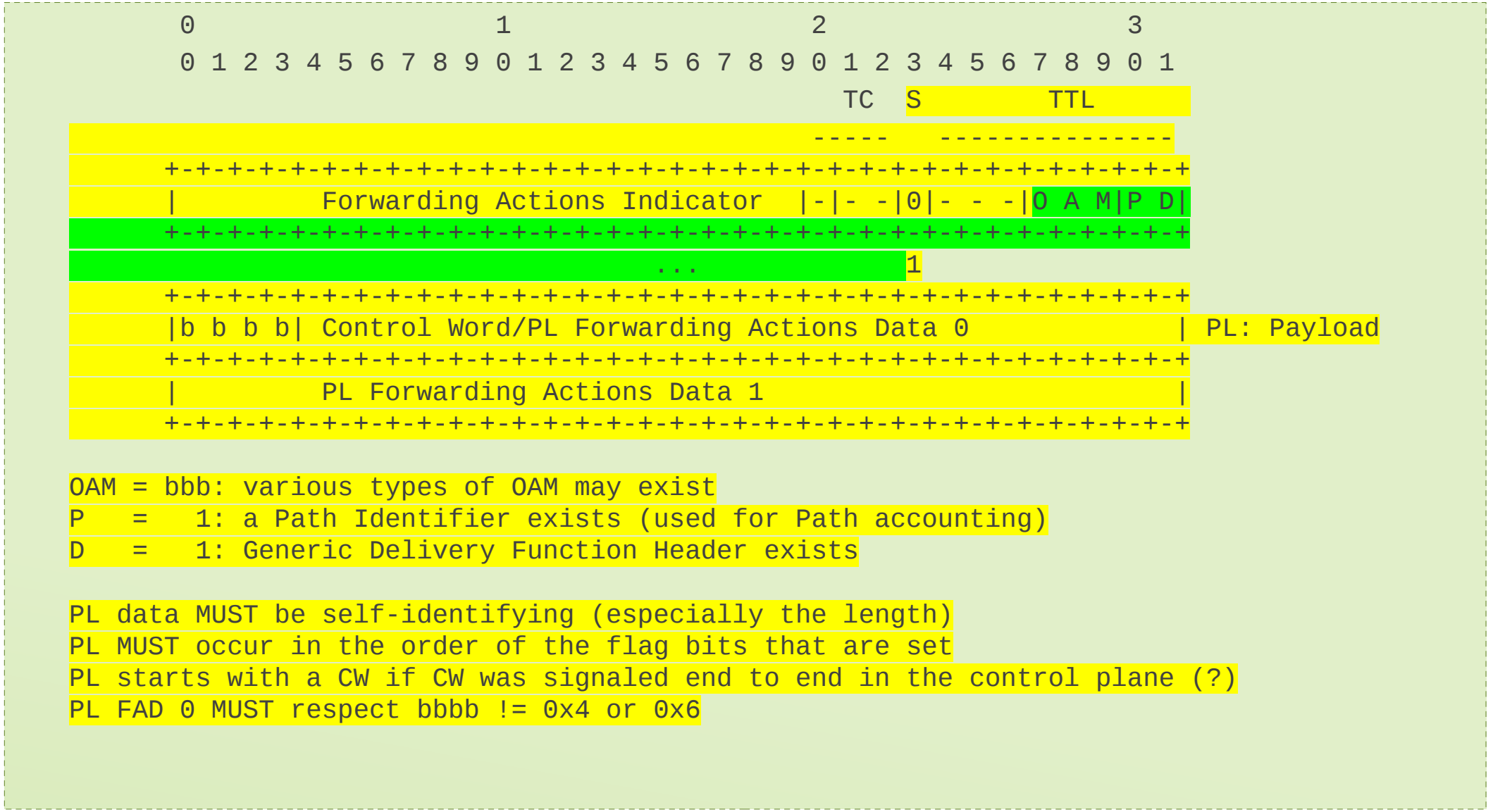
Alternatively, the repeated FAI can set the GE correctly, at the cost of repeating the GISS/EL

Needs more thinking

© 2021 Juniper Networks

Juniper Business Use Only

How the PL FAD Flag Bits Are Used



© 2021 Juniper Networks

Juniper Business Use Only

References

Entropy Label: RFC 6790

GISS: https://tools.ietf.org/html/draft-bestbar-teas-ns-packet-01

NFFRR: https://tools.ietf.org/html/draft-kompella-mpls-nffrr-01

Alt Mark: https://tools.ietf.org/html/draft-cheng-mpls-inband-pm-encapsulation-04 ; RFC 8321

iOAM: https://tools.ietf.org/html/draft-gandhi-spring-ioam-sr-mpls-00

Path Identifier: https://tools.ietf.org/html/draft-hegde-spring-traffic-accounting-for-sr-paths-02

Generic Delivery Header: https://tools.ietf.org/html/draft-zzhang-tsvwg-generic-transport-functions-00

© 2021 Juniper Networks

Juniper Business Use Only


Thank you
JUNIPER NETWORKS
Engineering
Simplicity