FORWARDING ACTIONS INDICATORS

Kireeti Kompella

IETF 112

draft-kompella-mpls-mspl4fa

DISCUSSIONS IN AND OUT OF THE OPEN DT REGARDING ISD AND PSD

1. When should some indicator/data be in the ISD vs in the PSD?
2. What should be said about the PSD in the ISD
3. How should extensions of the indicators be handled?
4. What about standard indicators that are not understood by someone?
5. What about "non-standard" indicators?
   - Accommodating "user-defined" (provider-defined) indicators and data fields would be friendly and powerful

1. ISD

Philosophy:

- In-stack data must be processed with some urgency
  - if not, this indicator and data shouldn't be part of ISD
- ISD must be coded compactly, and processed fast
- ISD must follow the label format (BoS MUST be respected)

2A. PSD

Philosophy

1. PSD need not be ultra-compact, nor does it have to fit into label fields, nor respect the BoS bit.
2. PSD should be self-describing. A TLV-type approach may be reasonable.

Given (2), too much detail in the ISD is redundant, and can lead to confusion

- If ISD says field X is present, but X is not in PSD, what should be done?
- If ISD says field X is absent, but X is in PSD, what should be done?

2B. SUGGESTION

- Indicators only say who should look at PSD
  - An optimization, where the "worst case" is that every hop looks at PSD
  - This means scanning the entire label stack (!)
- Use two bits:

00: no one needs to look at PSD

01: every node MAY look at PSD, but only egress MUST

10: every hop SHOULD look at PSD

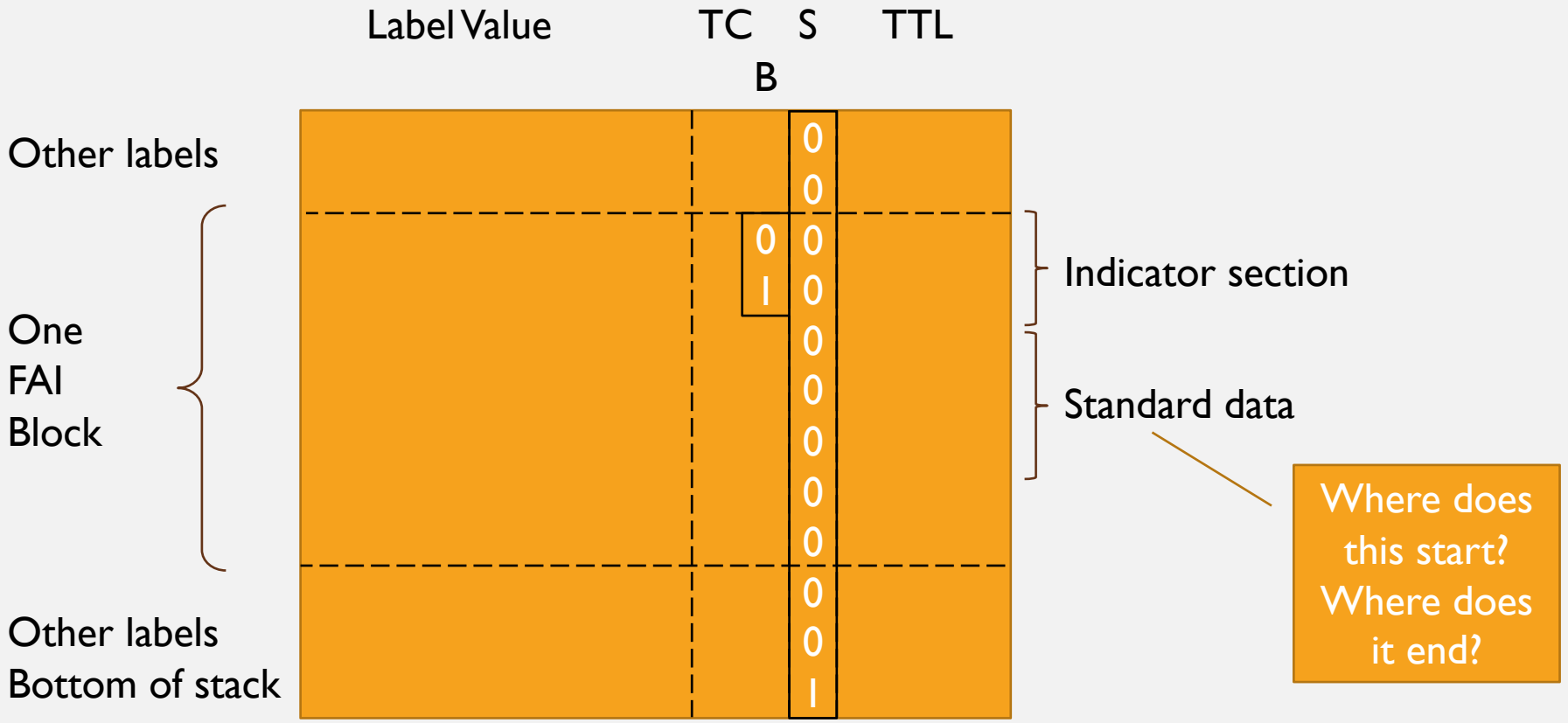
11: every hop MUST look at PSD; a node that cannot must drop packet

FAILSAFE: egress will have popped entire label stack; it can tell whether there is PSD (see previous presentation); if so, it can parse and process it

3A. HANDLING EXTENSIONS

- Use a bit, the E bit to indicate that the indicators continue
  - When E = 0, indicators end
- Standard data fields start after indicators are done
- If you invert this, there is an "end of indicators" bit
  - This matches the "Bottom of Stack" logic
  - When S = 1, labels end

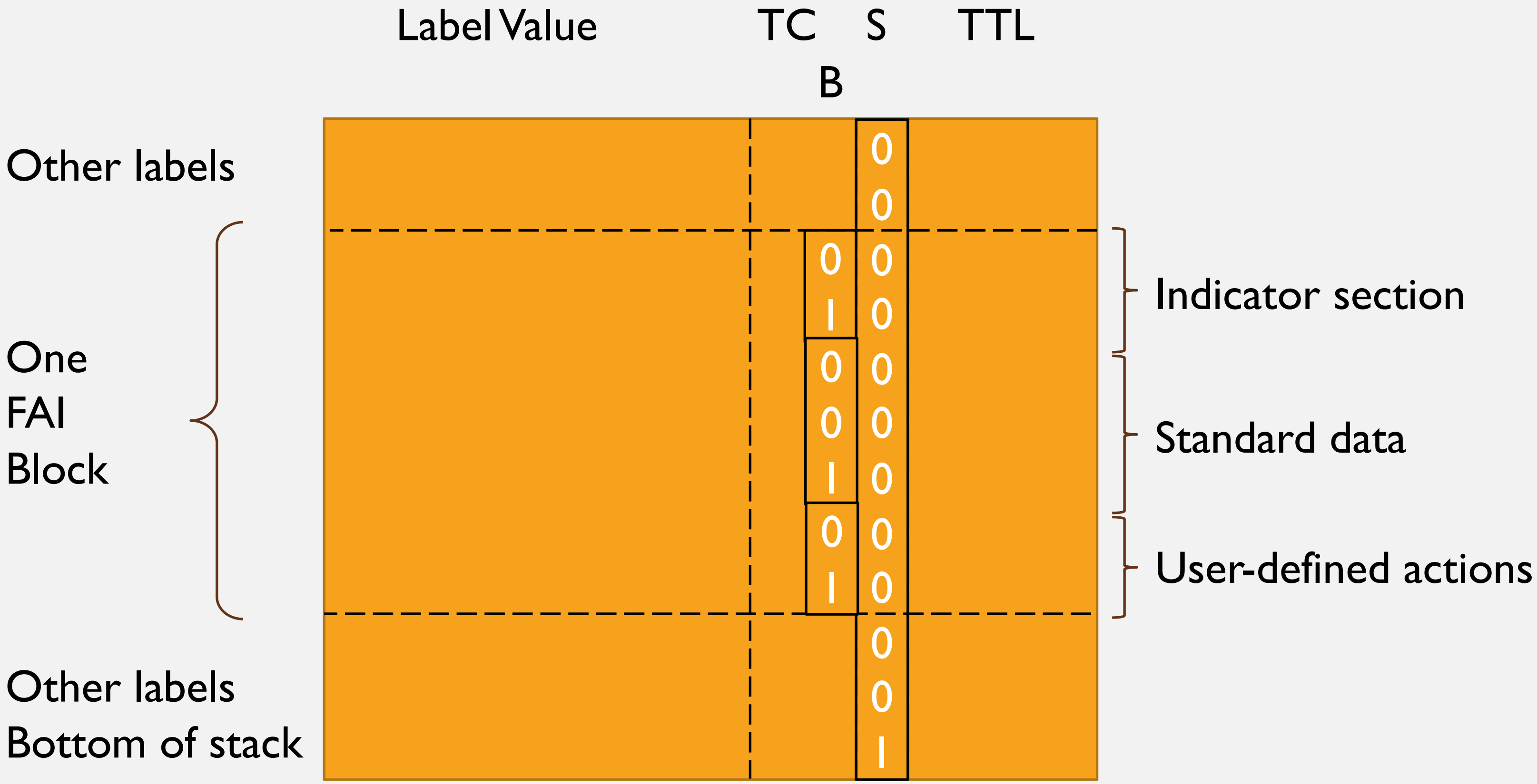
3B. CURRENT: "Bottom of Indicators" BIT (if you invert the E bit)



4. WHAT ABOUT STANDARD DATA?
5. WHAT ABOUT USER-DEFINED DATA?

- So, what if 50 standards data fields are defined, but an implementation only understands 5 of them? How does it know when the standard data is over?
- If there is user-defined data, how can an implementation know where it starts and where it ends?
- Solution: redefine the "bottom of indicators" bit for all these purposes
  - Think of this as the "bottom of section" bit

3-5. PROPOSAL: "Bottom of Section" BIT (modeled after "BoS" bit)



IMPLEMENTATION X THAT ONLY UNDERSTANDS SOME FIELDS