# Service Function Chaining (SFC)
# An Overview and IETF Work

Adrian Farrel : Old Dog Consulting

<adrian@olddog.co.uk>

India Internet Engineering Society (IIESoc)
Connections : A pre-IETF India Forum, October 31st - November 1st, 2018

# Menu

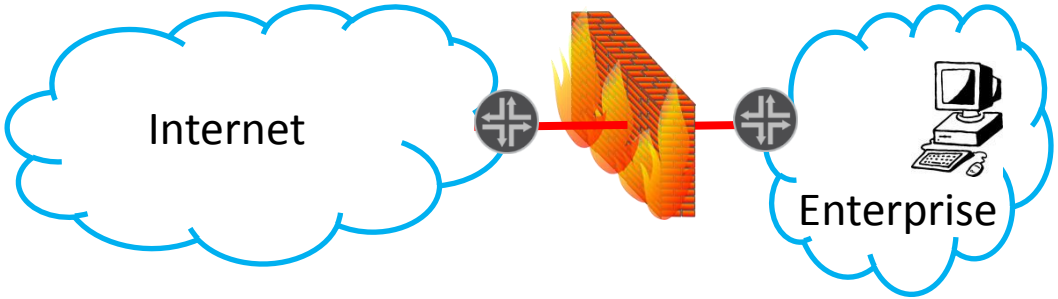
- What is Service Function Chaining (SFC)?
  - How is it done today?
- What work is the IETF doing?
  - Terminology and Architecture
- The Network Service Header (NSH)
  - An encapsulation for SFC
- System Components, Processing Rules, Clever Stuff
  - Meta Data
  - OAM
- Other approached
  - L3VPN, MPLS, Segment Routing
- Control Plane requirements and solutions
- References

# Why me?

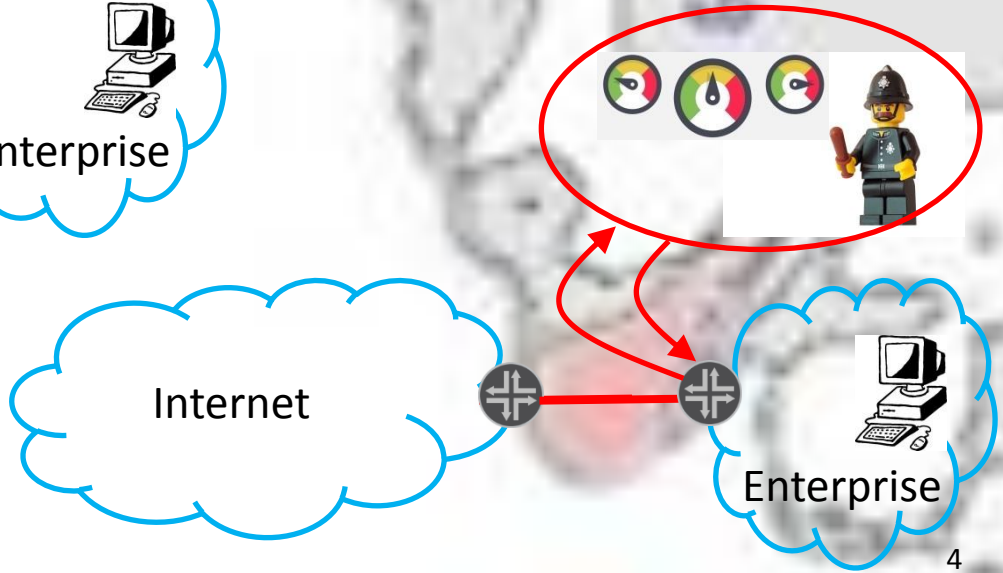
- Participating in IETF SFC standardisation via:
  - SFC working group
  - BESS working group
- Served as IETF Routing Area Director
  - For 6 years up to March 2015
  - Chartered the IETF's SFC activity
- Enthusiastic supporter of increased IETF participation from India
- I write books of fairy stories so I am uniquely qualified

# Classic Service Function Delivery

- "Bump in the wire"
  - Historically implemented as a dedicated device
  - Sits as an "invisible" feature on a link




- Port-attached local device
  - A configuration feature of a gateway (e.g. CE)
  - Port-mapping means function is also "invisible"

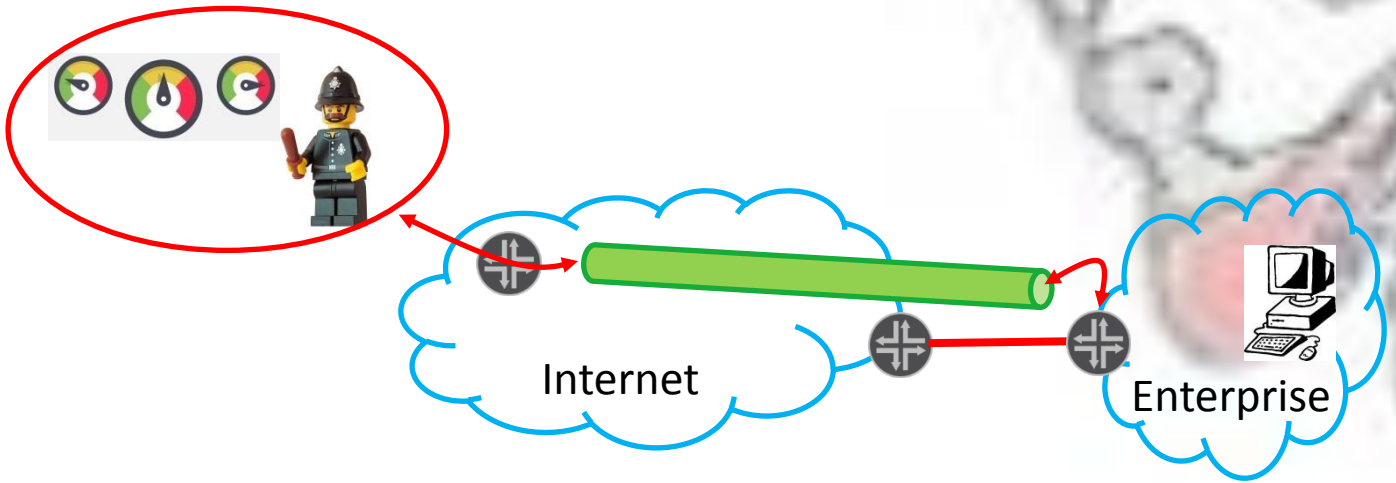


# Limitations of Classic Service Functions

- Many limitations
  - Large volume of under-used devices
  - Multiple devices to provide a set of functions
  - Upgrades and new functions are hard
    - At best need software upgrade at every site
    - May need physical visits
  - Management is highly distributed
    - Remote (or sometimes local!) login to every device
    - High chance of mismanagement

# Off-Path Service Functions

- Service function is located somewhere remote
- Packets are “seamlessly” extracted at the gateway and tunnelled to the service function
- A “remotely provided locally attached” approach
- Obviously, not a perfect technique for load reduction or security



# Introducing the Data Centre

- Data centres allow service functions to be virtualised
  - Placed off-path
    - Means traffic has to be routed (tunnelled) to them
  - Achieves cost-effective scaling
    - One service function instance can serve many traffic flows
  - Achieves flexible scaling and load balancing
    - New service function instances can be spun up easily
  - Highly agile
    - New functions and new versions of functions can be rolled out
  - Simple management
    - Service functions are “local” to the management application and consistent
  - Build sophisticated sequences (chains) of service functions

# So What are the Real Use Cases?

- Most of the actual use cases on the table are quite simple
  - Firewall
    - Just divert to the firewall then return to path
  - Load balancers
    - Divert macro flows to a function that distributes sub-flows across links or ACs
  - TCP Proxy
    - Add function in the TCP payload
    - Terminate the session, do function, start new session
  - End-point Selection
    - Pick a network exit point (such as for dual-homed CEs)
    - See also load balancers
- None of these is a complex or a long chain of functions
  - That may mean that everything that follows is over-engineering!
- One more complex example ***might*** exist for mobile connectivity
  - Derive caller ID, apply billing (per call, per byte), call-based access controls, parental controls, firewall
  - But isn't that already done in the wireless realm?

# IETF's SFC Work

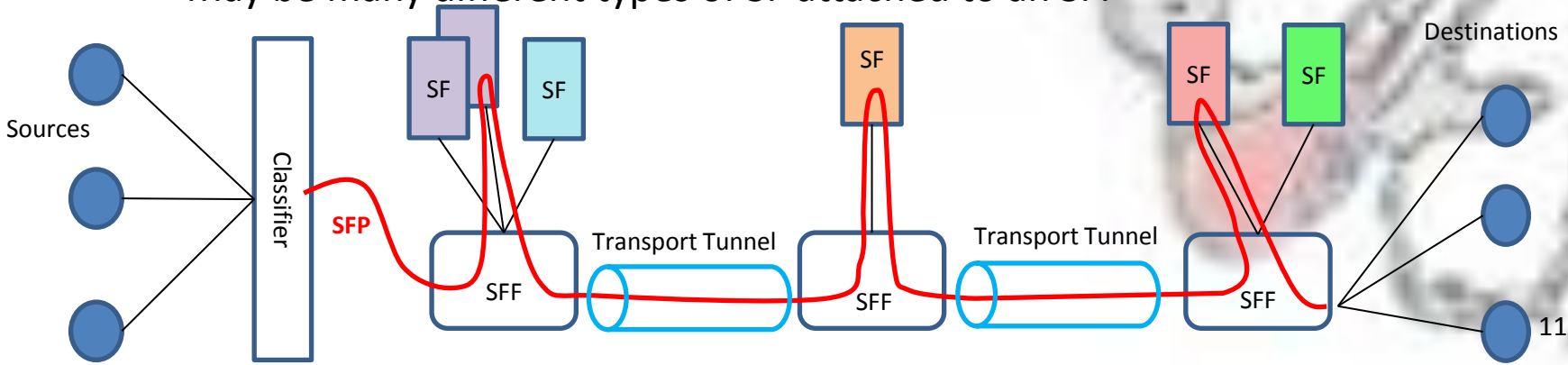
- SFC Working Group
  - https://datatracker.ietf.org/wg/sfc/about/
  - Formed September 2013
    - I was the chartering AD
  - Chairs
    - Jim Guichard (james.n.guichard@huawei.com)
    - Joel Halpern (jmh@joelhalpern.com -- Ericsson)
  - Five RFCs so far
    - RFC 7498 – Problem Statement
    - RFC 7665 – Architecture
    - RFC 8300 – Network Service Header (NSH) encapsulation
    - RFC 8393 and RFC 8459 – Minor extensions to the NSH
  - Most interesting work in progress
    - Service Function Chaining Use Cases in Mobile Networks
    - https://datatracker.ietf.org/doc/draft-ietf-sfc-use-case-mobility/

# IETF SFC Terminology

- Service Function (SF)
  - A function that is responsible for specific treatment of received packets
  - Can be realized as a virtual element or be embedded in a physical network element
  - Can act at various layers of a protocol stack
  - E.g., firewalls, WAN acceleration, application acceleration, DPI, server load balancers, NAT44, NAT64, HTTP header enrichment, TCP optimizers
- Service Function Chain (SFC)
  - An ordered (or partially ordered) set of abstract service functions
  - Applied to a set of packets, frames, or flows
- Service Function Path (SFP)
  - A specific instance of an SFC
  - Allows control of specific instances of SFs
  - Applied to a subset of packets/frames (usually whole sub-flows)
- Service Function Forwarder (SFF)
  - A device that forwards traffic along an SFP for processing by the next SF

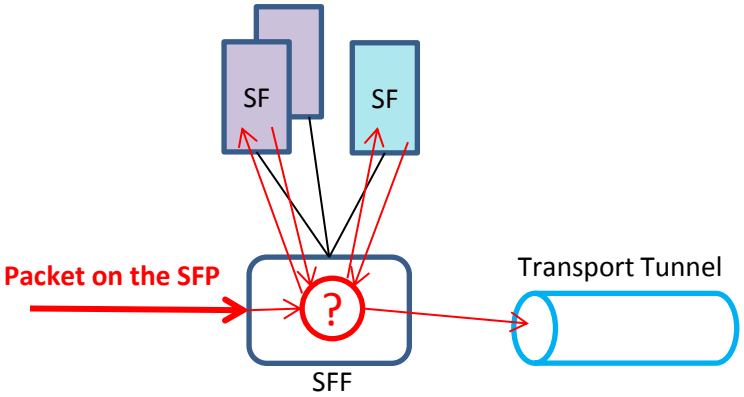
# The IETF's SFC Architecture

- Packets arrive at a Classifier
  - Matched against flow definition criteria
  - Placed onto a specific SFP
- Packets are tunnelled between SFFs
  - Tunnels depend on local technology
  - SFFs and tunnels form an overlay network
- SFFs deliver packets to locally instantiated SFs
  - May be many instances for load balancing
  - May be many different types of SF attached to an SFF



# Requirements for Packet Encapsulation

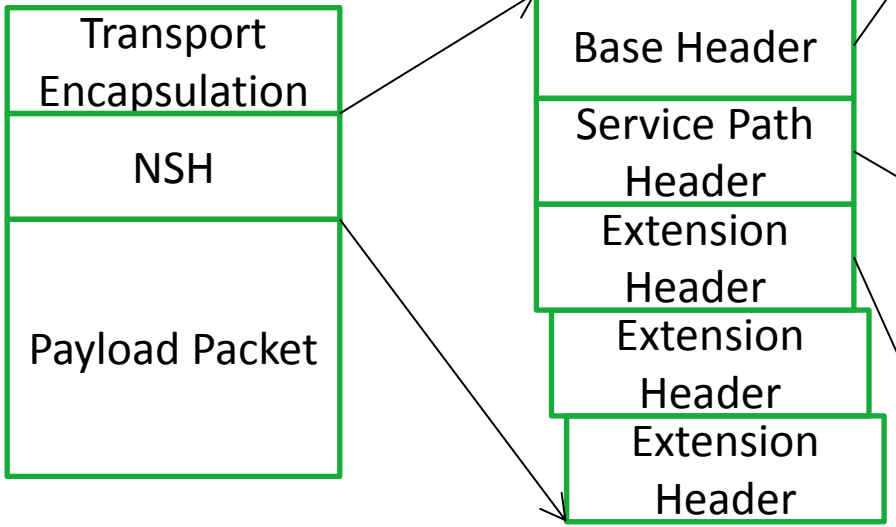
- SFF has to know
  - To which local SFs to deliver a packet (and in what order)
  - Any criteria to help choose between multiple instances of SFs
- SFF has to know out of which transport tunnel to send the packet (toward the next SFF)
- Need loop prevention
- Do not want to perform classification at each SFF
  - It is expensive and can delay packets



# The Network Service Header (NSH)

- SFC encapsulation should be layer agnostics
  - Should not know/care what the payload is
  - Should not know/care what the transport is
- Light-weight but fully functional
  - Address the requirements on previous slide
  - Enable complex SFPs
  - Simplify SFF and SF implementation

# NHS Encoding (RFC 8300)




| Ver | | TTL | Length | | MD Type | Next Protocol |
|---|---|---|---|---|---|---|

- Version (0x0)
- TTL counts down from 63
- Length (in long words) of whole NSH
- Meta Data type (see later slide)
- Next Protocol (Protocol type of next header)
  - IPv4/IPv6/MPLS/Ethernet etc.

| Service Path Identifier (SPI) | Service Index |
|---|---|

- SPI indicates the specific SFP in use
- SI indicates which SF to process next

- One or more Extension Headers
- Used principally (only?) to carry Meta Data
  - See later slide

# Classifier Processing Rules

- Classifier
  - Receives packets
  - Determines to which SFP a packet belongs
  - Applies an NSH
    - Sets SPI to indicate the SFP
    - Sets SI to indicate first SF on the path
    - Sets TTL “appropriately”
    - Sets Next Protocol to identify payload
  - Selects tunnel towards first SFF
    - Applies transport encapsulation
  - Sends packets

# SFF Processing Rules

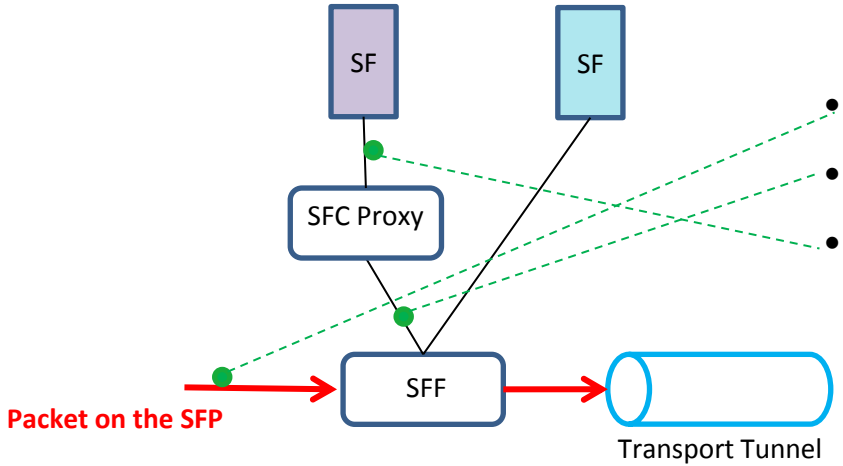
- SFF
  - Receives packets on a transport tunnel
    - Does the tunnel identify the SFP?
      - No, that would not scale: many SFPs may traverse a pair of SFFs
    - Strip the transport header
  - Do NSH TTL processing
    - Decrement and discard if goes to zero
  - Use SPI/SI to index the SFI to process the packet
    - Packet is for local processing
      - Pass packet (with NSH) to SF [see next slide]
      - Receive packet back from SF
      - Loop
    - Or
      - Select transport tunnel towards next SFF
      - Impose transport header
      - Send the packet

# SF Processing Rules

- SF
  - Receives packets from the SFF
  - May check the NSH
  - Steps over the NSH and acts on the packets
    - Result may be:
      - Packet OK to continue
      - Packet modified
      - Packet quarantined or sent somewhere else for processing
      - Alarm raised
      - Packet discarded
      - Etc.
  - NSH updated
    - SI decremented to indicate next SF to use on the SFP
  - Packets returned to (same) SFF

# SFC Proxy

- Suppose you have a legacy SF
  - It uses port attachment
  - It expects native packets
  - It cannot recognise an NSH
- The SFC Proxy strips the NSH and keeps per-port state
  - Forwards the native packets to the SF
- SFC Proxy receives packets back from the SF
  - Re-imposes NSH
  - Decrements SI




- Packet with Transport Header and NSH
- Packet with NSH
- Raw packet

# Meta Data

- What is Meta Data?
  - Information about the packet that is carried along with the packet
    - May be derived from the packet (e.g., hash or DPI)
    - May be generated by an SF (e.g., caller ID or content type)
  - Used by SFs to help execute their functions on the packet
    - Generally, Meta Data ***could*** be regenerated by an SF, but would be wasteful of processing and configuration
- Where do you draw the line?
  - A Classifier works on a packet to select the SFP
  - That work is carried in the NSH as the SPI
    - The SPI is not considered to be Meta Data

# Meta Data Use Cases

- Use cases for SFC Meta Data in IP networking are a little unknown
  - Is this a hammer looking for a nail?
  - For broad classifications maybe just use a different SFP (hence SFI) since that is cheap
  - Is just trying to solve a layer 5 problem at layer 3?
  - Do you really need this information on every packet?
    - Per packet Meta Data (such as the SPI)
    - Per flow Meta Data (applies to all packets with the same 5-tuple)
    - Per SFP Meta Data (applies to all packets on the same SFP)
- But for a discussion of possible Meta Data requirements see:
  - https://datatracker.ietf.org/doc/draft-ietf-sfc-use-case-mobility/

# Meta Data Encoding

- NSH uses an Extension Header to carry per-packet Meta Data
- The MD Type field in the Base Header indicates
  - Type x1
    - A fixed-length 16-byte Extension Header is present
    - Format and content is “context-specific”
      - Means SF already knows what to expect to see
      - SFP must be built only from SFs using the same format
  - Type x2
    - Variable length Extension Header
    - Contains its own class, type, and length fields
    - Bandwidth and parsing challenges arise!

# Reclassification and Other Clever Stuff

- A Classifier may be placed anywhere on an SFP
  - Allows choices to be made
    - Forks
    - Loops
    - Jump-over
  - Choice can be based on packet header/content and metadata
- In practice, Classifier may be co-located with an SF, SFF, or SFC Proxy
- For example
  - Simple SFC sends all packets to a Firewall
    - Good packets are forwarded : Bad packets are dropped
    - Suspect packets are moved onto a different SFC to be analysed and logged
      - Packets that are good after all, are forwarded
      - Packets that are bad and newly bad are sent onto a new SFC for security measures to be triggered

# OAM?

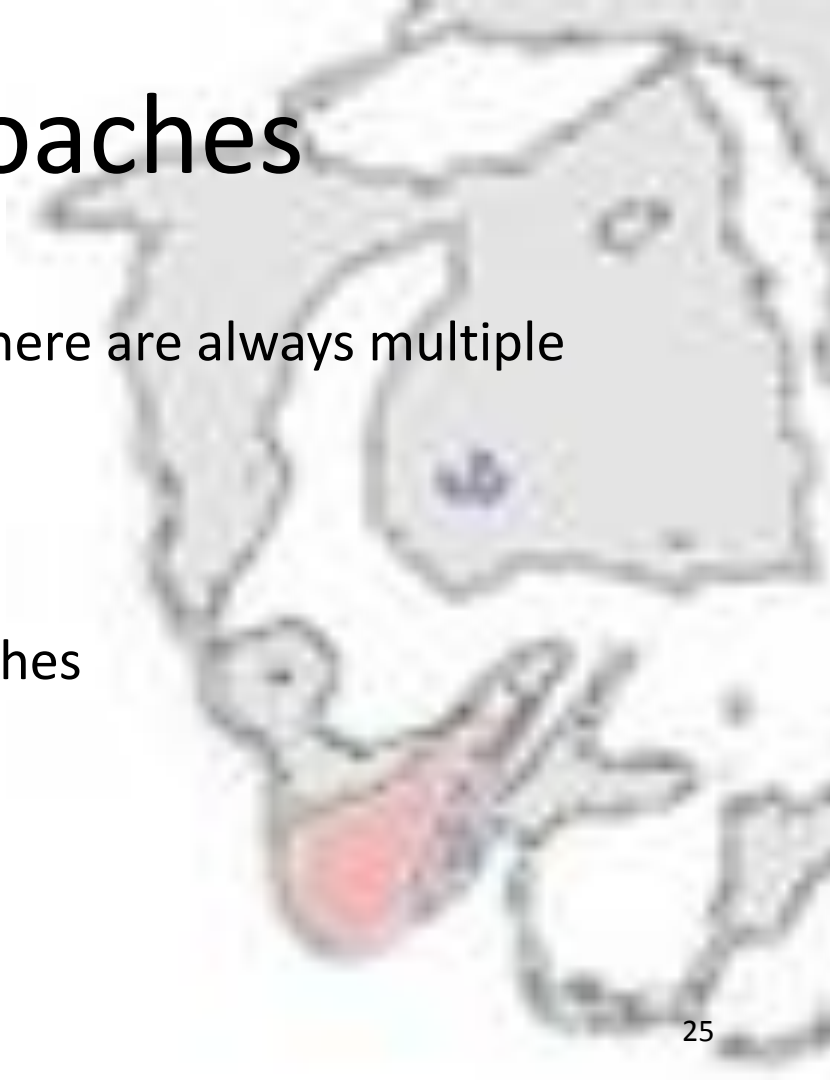
- We want to know
  - Is an SFP up (end-to-end)?
  - Which SFs are on the SFP and in what order?
  - Did a packet execute the desired SFs and in what order?
  - Which specific SFIs did a packet transit and in what order?
- What can we do with injected packets?
  - Does an SF have to be OAM-aware?
  - They don't tell us how the real packets are processed
- What about "inband OAM" (iOAM)
  - We can use Meta Data to record the path of a packet
  - Ugly amount of data
  - Probably needs hop-by-hop digital signatures
  - Maybe the SFF can do the work (saving the SF from the pain)
- IETF work (so far) is inconclusive
  - It is sad when OAM is not built into a new technology from day one

# Drawbacks to NSH

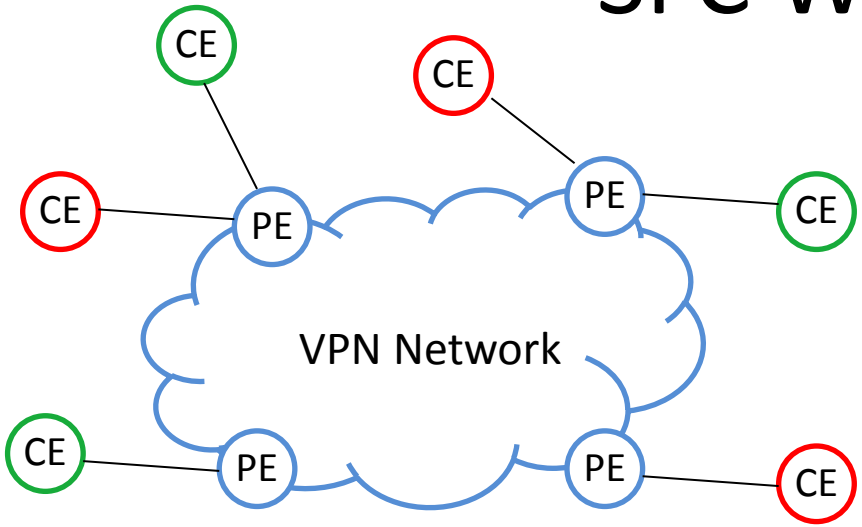
- There are some concerns about the NSH
  - SFFs should be able to process packets at line speed
    - Do they need new silicon for a new encapsulation?
  - Legacy SFs need an SFC Proxy to strip the encapsulation
    - An SFC Proxy is not so simple unless it is one per SFP
    - And, anyway, who will make/sell proxies?
  - In-packet Meta Data makes hardware processing hard
    - Also may make MTU prediction difficult
  - Lack of defined OAM
    - Early hardware will ship without OAM support
  - Rule to decrement SI at the SF is “clumsy”
    - Would like to support gaps in the SI sequence so that SFPs can be modified in place
    - Can be handled by programming Classifiers within SFs

# Other Approaches

- Of course…
  When a problem space is identified, there are always multiple solutions proposed
  - This is a good thing!
  - Industry can test and experiment
  - “Let the market decide”
- A quick visit to three possible approaches
  - L3VPN
  - MPLS
  - Segment Routing

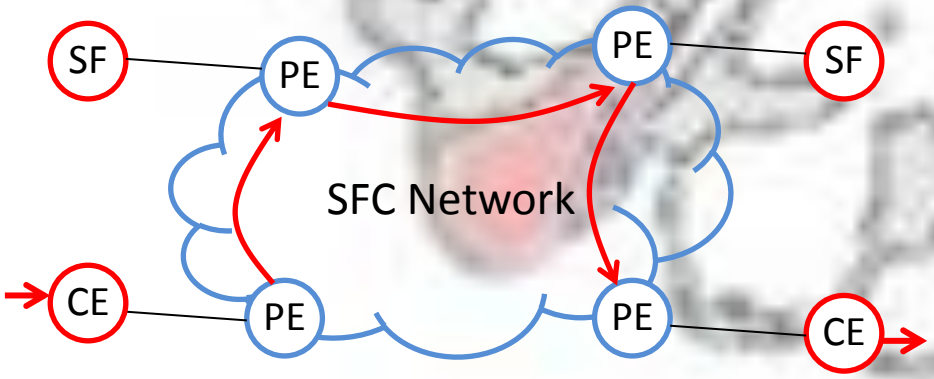


# SFC With L3VPN



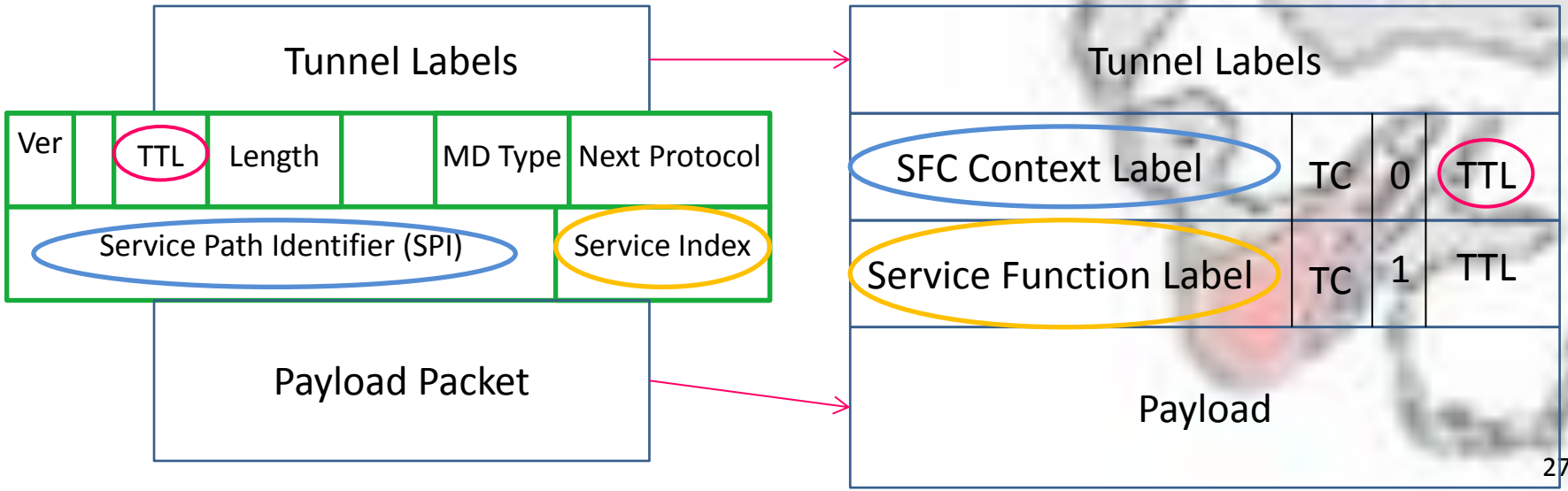

- Supports multiple instances
  - Indexed by VPN label
- Outbound traffic
  - Routed to remote PE
  - Based on VRF populated by BGP
- Inbound traffic
  - Port-based attachment to CE

- Supports multiple SFPs
  - Indexed by VPN label (i.e., SPI)
- Outbound traffic
  - Source PE “classifies” traffic
  - Routed to next PE (SFF)
  - Based on SFP populated by BGP
- Inbound traffic
  - Port-based attachment to SF



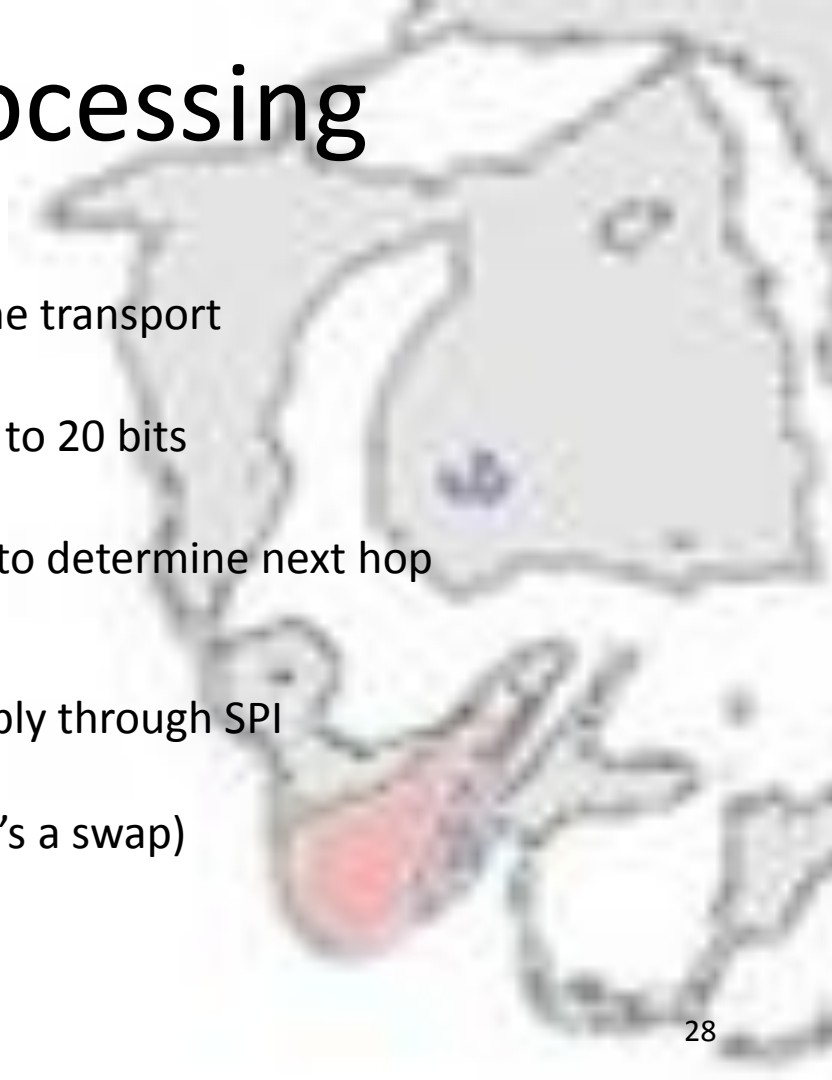
# SFC With an MPLS Data Plane

- There are lots of MPLS-capable routers
  - Perhaps they could act as SFFs without too much effort
  - Use a stack of two labels in place of the NSH (32 bits both ways)
  - Other fields “not needed”
  - Packets progress by label swapping



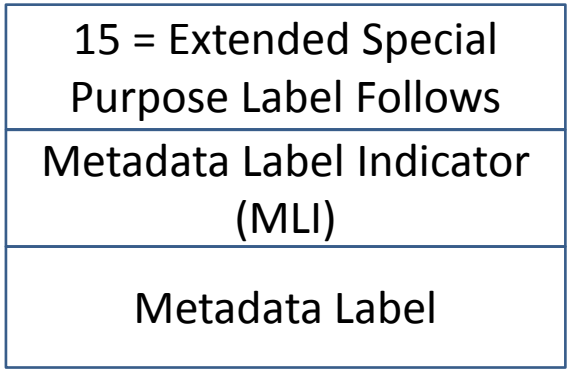
# MPLS SFC Processing

- Tunnels between SFFs “as normal”
  - Of course, we are interested in MPLS as the transport
- SPI and SI used “as normal” for NSH
  - Some limitation as SPI is constrained here to 20 bits
- MPLS-SFC processing…
  - Labels are looked up and acted on by SFF to determine next hop
    - Maybe forward to SFI or SFC proxy
    - Maybe forward to next SFF
  - In some cases action can be achieved simply through SPI
  - In other cases need the two label context
  - SI is updated before further forwarding (it’s a swap)
  - SPI and SI set during classification
    - Potentially also during re-classification



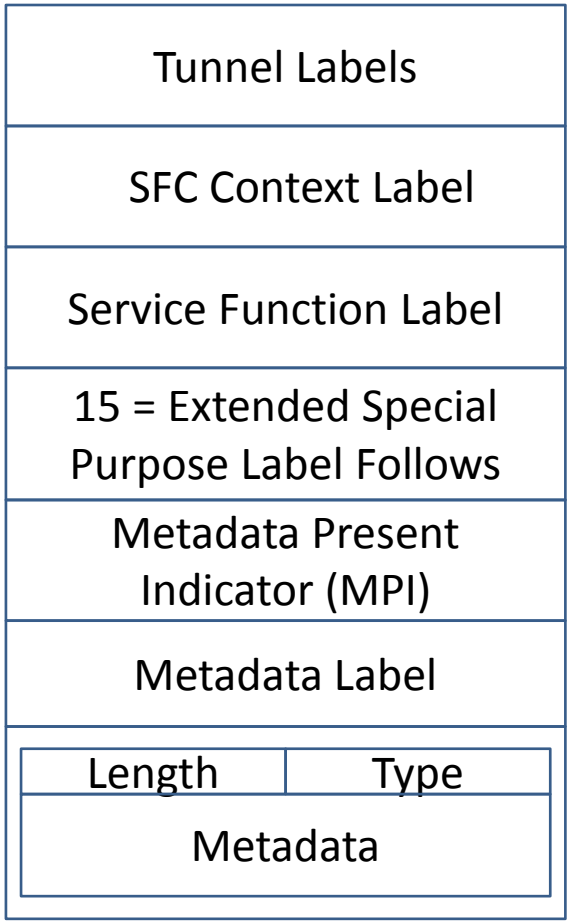
# What About Metadata?

- MPLS encapsulation not well suited for carrying “arbitrary” metadata
- We define an Extended Special Purpose Label




- This three-label sequence can be included at the bottom of the label stack
- Metadata label is an index into a store of metadata
  - Must also not use 0..15
- Store may be populated though management plane, control plane, or in-band (next slide)
  - This approach is not good for “per-packet metadata” (e.g., hashes)
  - Works fine for per-SFP or per-flow metadata

# In-Band Meta Data Distribution




- Send packets along an SFP without carrying payload (but still carrying metadata)
- Use an Extended Special Purpose Label
  - Hence, a three label sequence
- Metadata Label is the index for use in data packets
- Placed at the bottom of the label stack
- Rest of stack exactly as for SFP
- Meta Data carried as payload
  - Formatted as TLV
  - Type field defined by SFC WG for NSH
  - Meta Data as defined by SFC WG
- Nodes on the SFP can store the Meta Data

# Segment Routing for SFC

- Segment routing is a source-controlled, per-packet, traffic steering technique
  - A stack of hops is imposed on the packet and used in the network to control the route
  - Moves state from the network to the packet
    - No need for a signalling plane (still need routing/discovery)
  - Works for MPLS and IPv6
- Potential for application to SFC
  - The hops are the SFF/SFI pairs
    - Impose MPLS label pairs, or IPv6 SIDs
  - No need for SFP state in the network
  - Easy to vary the SFIs “on the fly”

# Control and Management Requirements

- How do I configure SFIs?
- Where are the SFIs?
  - How do I reach them? (Which SFF?)
  - What are their capabilities?
- Where are the SFFs?
  - How are they connected? (tunnels, addresses)
- How do I compute and “create” SFPs?
- What are the SFPs?
  - How does an SFF know which SFPs it serves?
  - How does an SFF know to which SFI to deliver packets?
  - How does an SFF know to which SFF to forward packets?
- How do I determine what traffic to put on an SFP?
- How do I tell the Classifier how to classify traffic to SFPs?

# Centralised or Distributed Control?

- A lot of this feels like a TE problem
  - Overlay network
  - Path computation
  - Traffic steering
- Achievable with a combination of central and distributed control
  - Planning
  - Configuration/instantiation
  - Discovery
  - Routing
- A role for something like a PCE?
- As demonstrated by L3VPN approach, routing is edge-to-edge
  - Makes BGP a good idea

# A BGP Control Plane

- Data plane agnostic (NSH, MPLS, …)
- Discovery
  - SFF announces locally attached SFIs
  - Route Target
    - Identifies the overlay network
    - Other nodes only import when the RT matched
  - Route Distinguisher (SFIR-RD)
    - Identifies this SFI advertisement
  - SF Type (SFT)
    - From the FCFS IANA registry
  - Controller(s) can know about available resources and locations
  - SFFs can know how to reach next SF on an SFP
- SFP Advertisement
  - Controller (or head end) announces each SFP
  - Route Target
    - So only participating nodes need to import the advertisement
  - Route Distinguisher (SFPR-RD)
    - Identifies the SFP advertisement
  - Service Path Identifier (SPI)
    - Uniquely identifies the SFP
    - Used in the forwarding plane to identify this SFP
  - Series of hops in the path each encoded as a Hop TLV (the specific SIs)
- BESS working group

# Resources

- SFC Working Group
  - https://datatracker.ietf.org/wg/sfc
    - SFC Architecture
      - https://www.rfc-editor.org/rfc/rfc7665.txt
    - Network Service Header (NSH)
      - https://www.rfc-editor.org/rfc/rfc8300.txt
- BGP Enabled ServiceS (BESS) Working Group
  - https://datatracker.ietf.org/wg/bess
    - Service Function Chaining using Virtual Networks with BGP VPNs
      - https://www.ietf.org/id/draft-ietf-bess-service-chaining
    - BGP Control Plane for NSH SFC
      - https://www.ietf.org/id/draft-ietf-bess-nsh-bgp-control-plane
- MPLS Working Group
  - https://datatracker.ietf.org/wg/mpls
    - An MPLS-Based Forwarding Plane for Service Function Chaining
      - https://www.ietf.org/id/draft-ietf-mpls-sfc
- Source Packet Routing in Networking (SPRING) Working Group
  - https://datatracker.ietf.org/wg/spring
    - Service Programming with Segment Routing
      - https://www.ietf.org/id/draft-xuclad-spring-sr-service-programming

Questions and Follow-up

adrian@olddog.co.uk