SOFTWARE DEFINED VIRTUAL NETWORKING SECURITY

CHAPTER 9 SERVICE FUNCTION CHAINING

Dijiang Huang, Ankur Chowdhary, and Sandeep Pisharody

OUTLINE

- Introduction
- Service Function Chaining Concepts
- SDN and NFV based SFC
- SFC Implementations
- Policy Aware SFC
- Secure Service Function Chaining (SSFC)

INTRODUCTION TO SERVICE FUNCTION CHAINING (SFC)

PROBLEMS WITH CURRENT MIDDLEBOX DEPLOYMENTS

- Middleboxes are devices used by network operators for performing network functions along the packet path from source to destination.
- Design of general purpose middleboxes for different network functions is an active area of research.
- Lack of proper deployment can lead to misconfigurations and increased operational expenses.

COMMON CAUSES OF MIDDLEBOX FAILURES

	Misconfiguration	Overload	Physical/ Electric
Firewall	67.3%	16.3%	16.3%
Proxies	63.2%	15.7%	21.1%
IDS	54.5%	11.4%	34%

PROBLEMS WITH CURRENT MIDDLEBOX DEPLOYMENTS

- About 9% administrators reported spending 6-10 hours/week dealing with middlebox failures.
- Adoption of new middlebox technologies is slow in the industry.

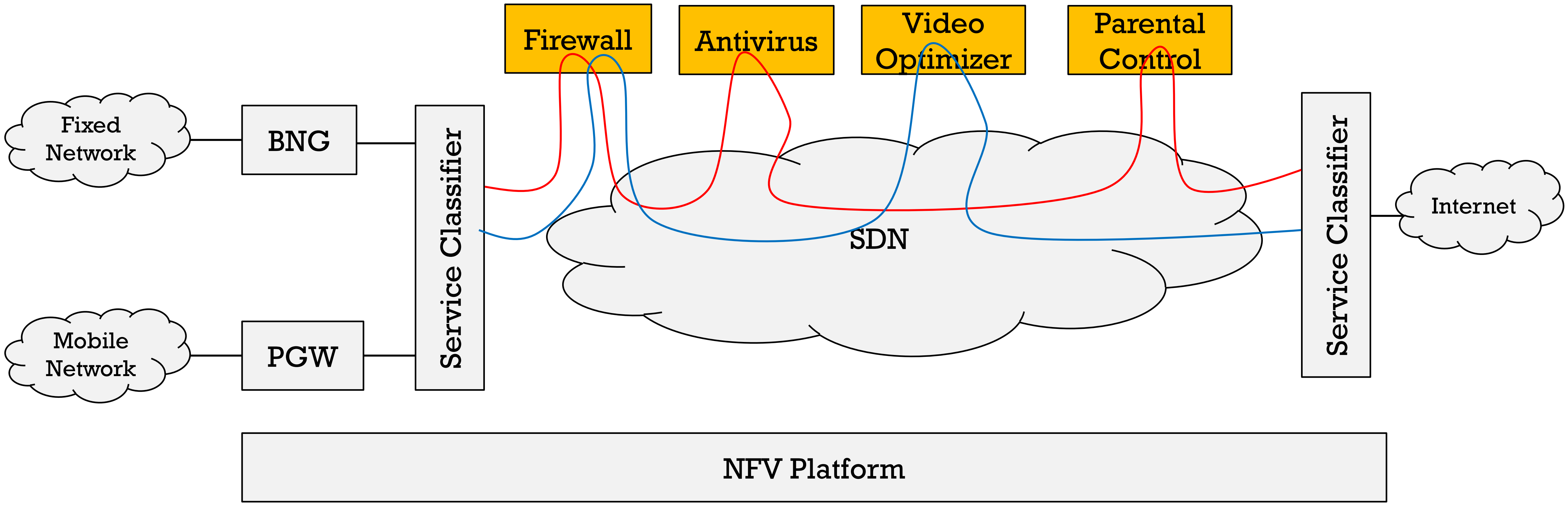
CHANGING CLOUD AND NETWORKING LANDSCAPE

- Shift from host-centric to data-centric model.
- Network services and computing resources available closer to the users.
- Need for flexible network function deployment models.
- Network Function Virtualization (NFV) enabled virtualized deployment of hardware components such as Routers, Firewalls, called Virtual Network Functions (VNFs).

SERVICE FUNCTION CHAINING

- Service Function Chaining (SFC) is the capability of creating connected chain of virtual network functions (VNFs) such as Firewall, Load-Balancer, Intrusion Detection and Prevention System.
- The service chaining helps in automated provisioning of the network applications with different characteristics.
- Example: A video or VOIP session has more resource demands compared to simple web-access.
- SFC ensures that specific applications are getting proper network resources or characteristics (bandwidth, encryption, QoS).

SERVICE FUNCTION CHAINING



SERVICE FUNCTION CHAINING

- On-demand provisioning of traffic sessions with user-specified requirements can be achieved using SFC.
- Surge in Internet-of-Things (IoT) and mobile devices make flexible (closer to user, adaptive) deployment of VNFs desirable.

SERVICE FUNCTION CHAINING ASSUMPTIONS

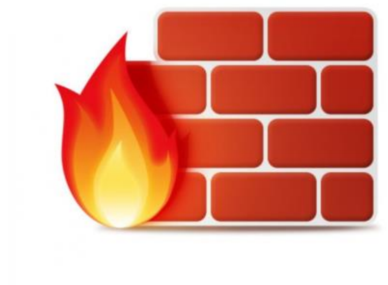
- Set of service functions in a given administrative domain may vary from time to time.
- The SFC policy and criteria of enabling particular VNFs is a decision local to the administrative entity of each service function (SF) domain.
- There is no global SFC logic. Each administrative entity can define its own SFC logic.
- Several SFC policies can be deployed simultaneously in order to achieve business objectives.
- SFC assumes independence of the underlying network setup.

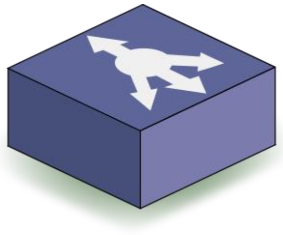
SERVICE FUNCTION CHAINING CONCEPTS

SFC Architecture, Challenges in SFC

NETWORK SERVICE

- An offering provided by the network service provider.
- Network service can be composed of single or multiple virtual network functions (VNFs).



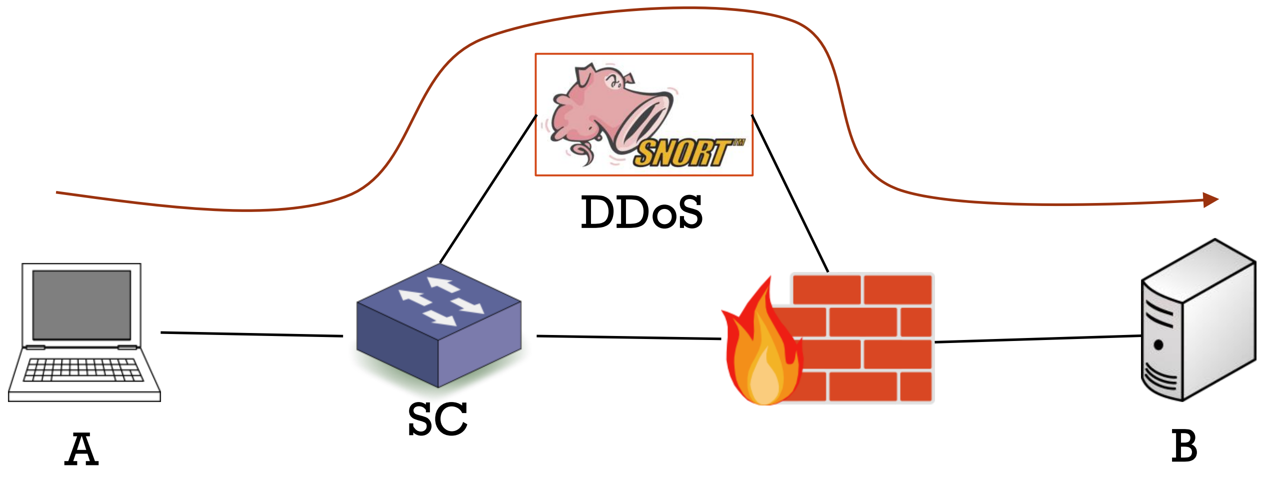


- E.g., Firewall, Load Balancer, IDS, etc.

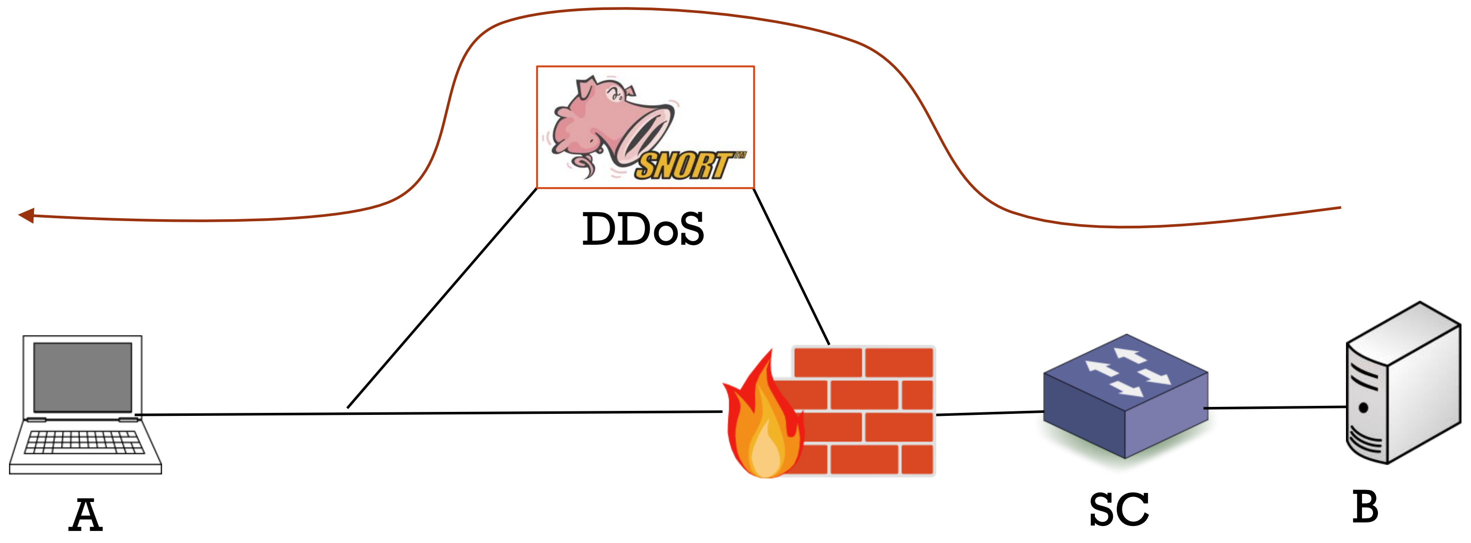
CLASSIFICATION

- Traffic flow segregation based on local policies defined for each segment of the network.
- Many security based service functions require bi-directional traffic of a connection.
- DDoS mitigation mechanism may require the visibility of return traffic to maintain proper state.
- Return traffic (server to client response) requires classification based on the forward traffic (client to server request).
- Element responsible for classification is known as Classifier or Service Classifier (SC).

CLASSIFICATION



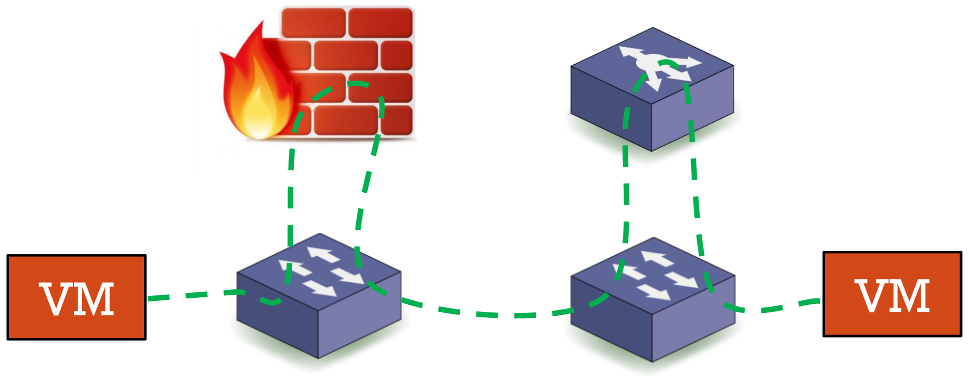
(1) Forward Flow A => B



(2) Return Flow A <= B

SERVICE FUNCTION CHAIN

- Ordered set of service functions that should be applied to the classified traffic.
- The order of application can be sequential or parallel based on network requirements.
- E.g., Firewall and IDS should be used in serial order, whereas Monitoring and Firewall functions can be parallelized.



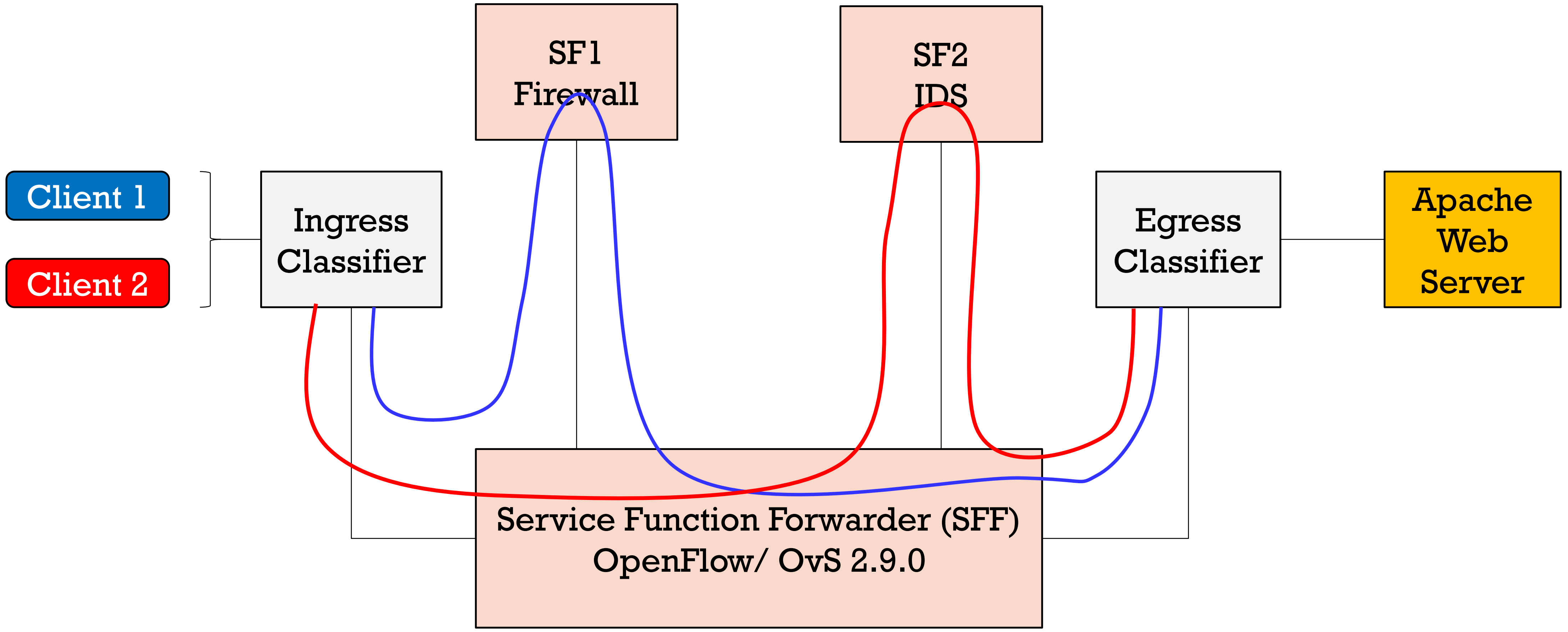
SERVICE FUNCTION (SF)

- A service function used for treatment of traffic in a specific way.
- A service function provides value added service to the traffic flows.
- A service function may perform its function(s) at one or more OSI layers.
- Some examples of service function include TCP optimizer, load-balancer, NAT, etc.
- A virtual function at the network level is known as Virtual Network Function (VNF), e.g., virtual Firewall, virtual Router.

SERVICE FUNCTION FORWARDER (SFF)

- Physical or virtual device used fir forwarding the traffic to the neighboring section of the network based on SFC-encapsulation policy.
- SFF handles both ingress and egress traffic for a network segment.
- SFF receives packets/frames carrying SFC header information and forwards the packets/frames to the associated SF instances using header information.

SERVICE FUNCTION FORWARDER (SFF)



SERVICE FUNCTION PATH (SFP)

- A constrained specification of packet traversal according to the service function chain specification.
- The granularity of SFP can be exact location or it can be less specific.
- SFP allows network operators to have a certain level of control over the selection of SFF/SF.
- Think of service chain as the "intent"; service path the actual instantiation of the chain in the network.

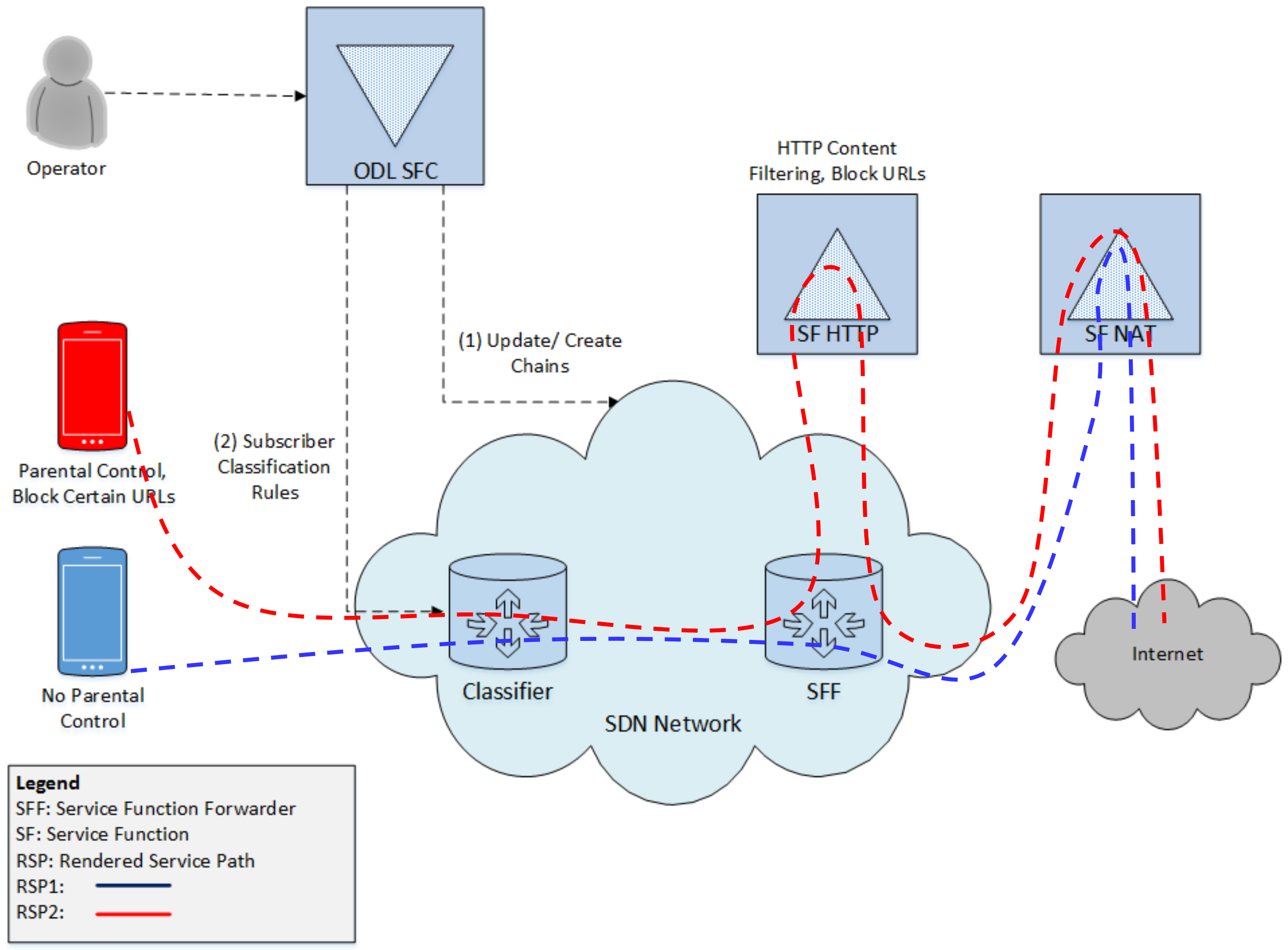
SFC ENCAPSULATION

- SFC encapsulation consists of information that helps in identification of SFP for SFC aware traffic.
- The encapsulation information may also consist of metadata associated with data plane of the traffic being steered using SFC.

RENDERED SERVICE PATH (RSP)

- Rendered Service Path (RSP) refers to the actual path traversed by packets between two end-points in a network based on SFP specified by different network operators.

SFC EXAMPLE


Operator
ODL SFC
HTTP Content Filtering, Block URLs
SF HTTP
SF NAT
(1) Update/ Create Chains
(2) Subscriber Classification Rules
Parental Control, Block Certain URLs
No Parental Control
Classifier
SFF
SDN Network
Internet
Legend
SFF: Service Function Forwarder
SF: Service Function
RSP: Rendered Service Path
RSP1:
RSP2:

SFC EXAMPLE

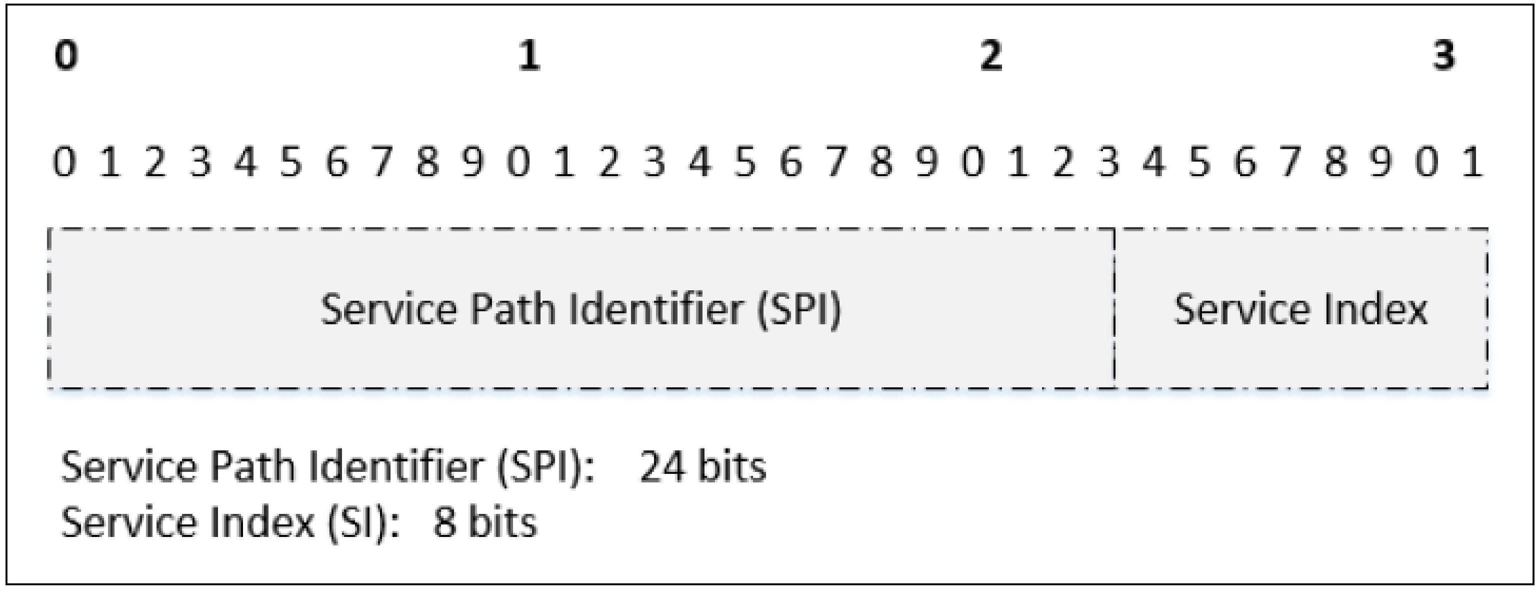
- The example showcases two different service function chains based on the application requirement.
- SFC parental control blocks web-traffic not deemed appropriate for the children.
- The classifier steers the traffic through HTTP and NAT service functions and filters inappropriate traffic at the NAT SF.
- SFC with no parental control steers the traffic through SF NAT on the path to the internet.

SFC EXAMPLE

- RSP1 shows the traffic path for the SFC with parental control.
- RSP2 shows the path traversed by the SFC with no parental control.
- Both service chains traverse through the SFF while accessing the services available on the internet.

NETWORK SERVICE HEADER (NSH)

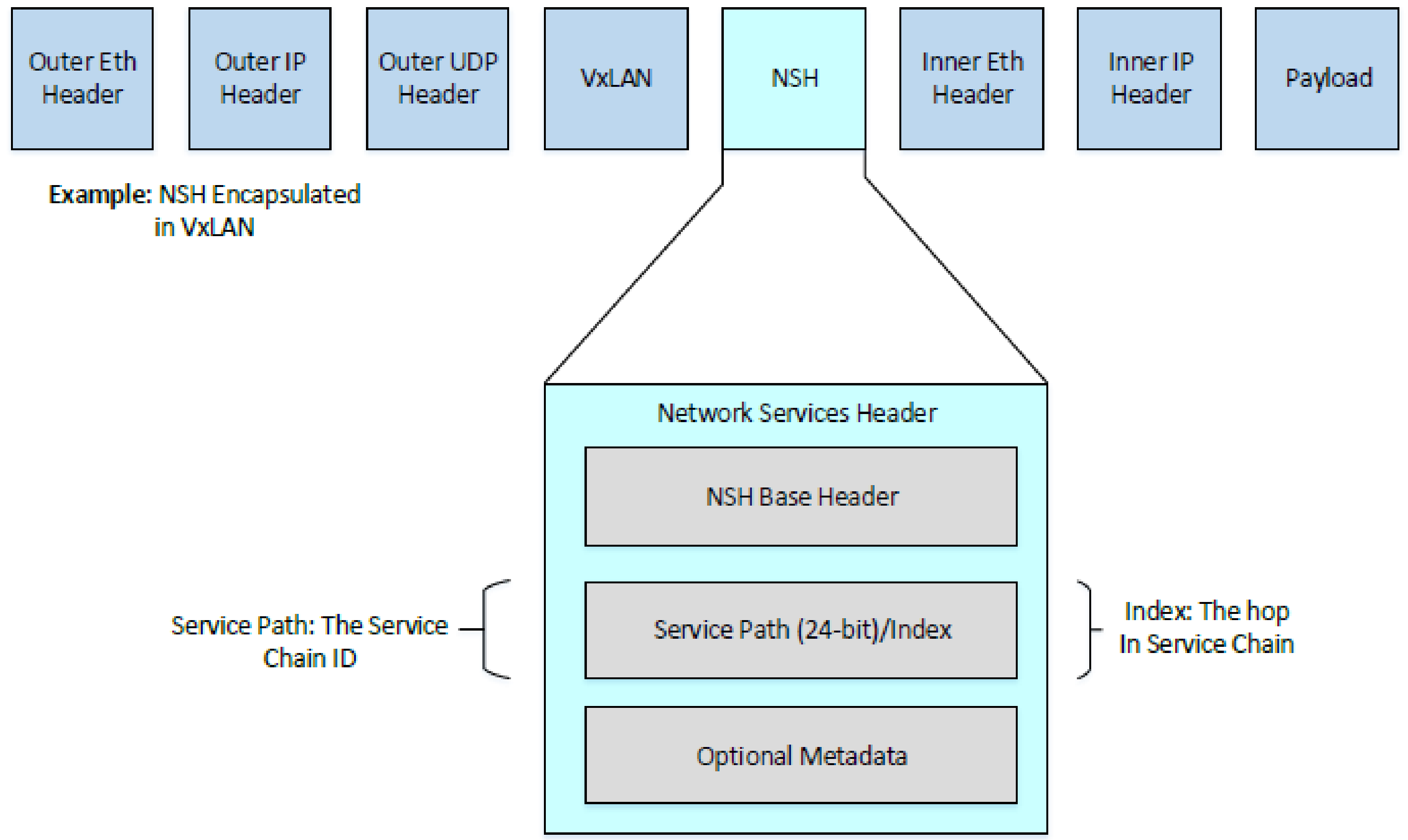
- Service plane protocol used for creation of dynamic Service Function Chain.


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
Service Path Identifier (SPI)
Service Index
Service Path Identifier (SPI): 24 bits
Service Index (SI): 8 bits

NETWORK SERVICE HEADER (NSH)

- Service Function Path Identifier (SFPI) – identifies the service function path.
- Service Index (SI) – identifies the location of service function path. Value decrements as SFs or service function proxy nodes are encountered.
- Default value for SI is 255.
- Optional metadata that can be shared between the participating entities and SFs.

NETWORK SERVICE HEADER (NSH)



NETWORK SERVICE HEADER

Advantages

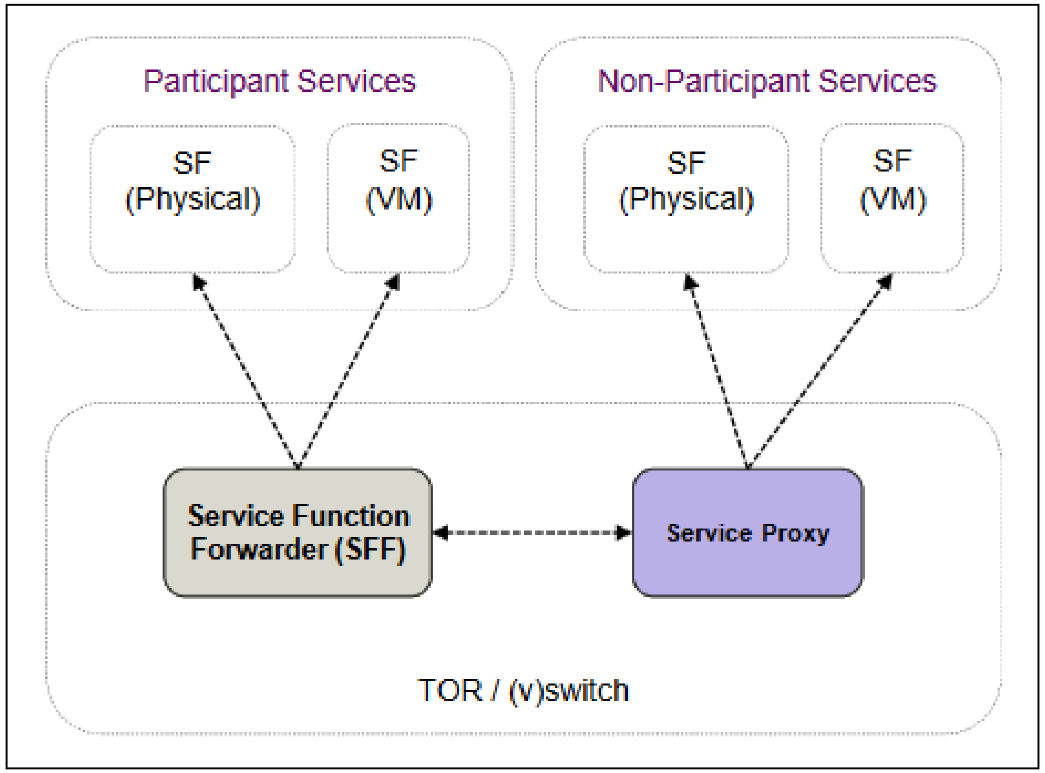
- NSH allows metadata to be sent as a part of SFC. This can be helpful in optimal service chain deployment.
- NSH supports wide variety of underlay and overlay protocols such as MPLS, GRE, VXLAN.
- NSH helps in simplifying the forwarding complexity due to complex topological dependencies

Limitations

- NSH currently has limited support in switches, kernels, and applications.
- SF should be aware of NSH being part of SFC.
- NSH encapsulation can introduce delays in the SFC.

SERVICE FUNCTION PROXY

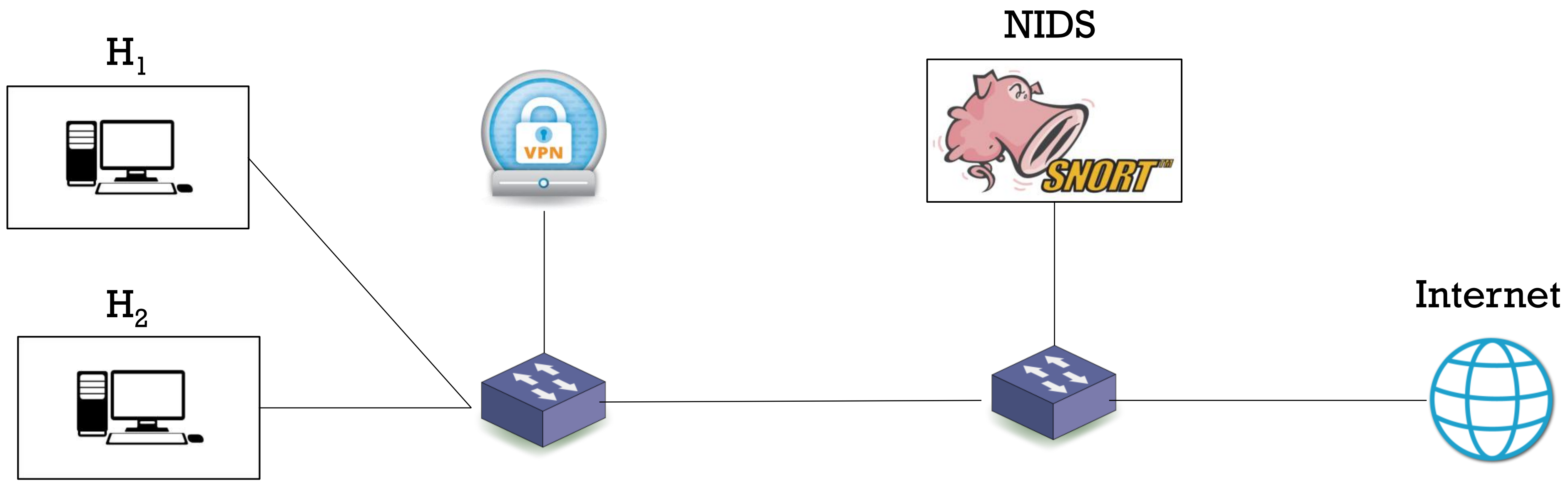
- Legacy service providers may not have the capability to process packets encapsulated with network service header (NSH).
- NSH architecture introduces the concept of service proxy for processing the network service headers and mapping to/from functions.



CHALLENGES IN SFC

- Topological Dependencies.
- Configuration Complexity
- Constrained Availability of SFs
- Ordering and Application of SFs
- Transport Dependence
- Elastic Service Delivery
- Traffic Selection Criteria
- Security Considerations

TOPOLOGICAL DEPENDENCIES



TOPOLOGICAL DEPENDENCIES

- The service delivery offered by SFC is often tightly coupled with the network topology.
- E.g. – Tenants communicating via fast tunneling networks like VLAN.
- Packet inspection modules such as Firewall and/or IDS along the service function path (SFP) can affect the service delivery capabilities.
- The scale, capacity and flexibility across the network is also impacted by the topological dependencies.

CONFIGURATION COMPLEXITY

- Logical and physical topologies are at times dependent on order of SFs in SFC.
- Changes in the ordering of SFs requires corresponding change in the physical or logical topology as well.
- Network operators are hesitant to make such changes due to fear of potential misconfiguration.
- The configuration complexity leads to static service delivery deployment.

CONSTRAINED HIGH AVAILABILITY

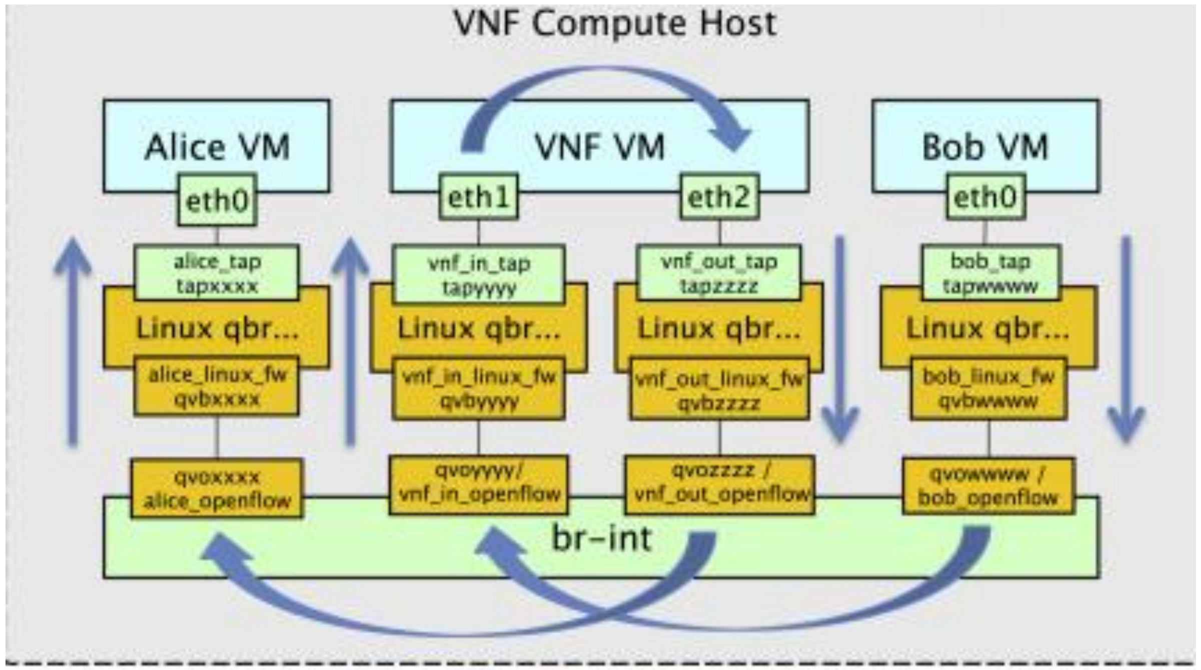
- Traffic reaches many service functions based on network topology.
- Alternate or redundant service functions must be placed in same topology as primary service functions.
- The effect of topological dependency is that the availability of service functions is constrained.

ORDERING AND APPLICATION OF SF

- Administrator has to consider the order dependencies between SFs.
- A standard way to enforce and verify the ordering is required.
- Currently there is no standard/protocol to enforce SFC ordering.
- The service policy application in SFC requires granular information for policy specification, which is currently not available in SFC deployments.

TRANSPORT DEPENDENCE

- SFC should be generic enough to support wide variety of underlay and overlay protocols such as Virtual eXtensible Local Area Network, Generic Routing Encapsulation (GRE) and MPLS.



ELASTIC SERVICE DELIVERY

- Adding and removing virtual network functions based on real-time traffic demands.
- This is difficult to achieve in existing network because of complex changes required for network topology and routing information.

TRAFFIC SELECTION CRITERIA

- Coarse grained traffic selection policies in existing SFC deployments.
- Policy routing and access control filtering can be used to achieve granular traffic selection in SFC.

SECURITY CONSIDERATIONS

- Service overlay is dependent on the transport protocols such as GRE, MPLS in the network.
- A secured protocol as IPSec should be selected for ensuring security requirements such as confidentiality or authentication.
- The classification policy used for selection of underlay or the overlay protocol should be trusted and accurate.

SECURITY CONSIDERATIONS

- SFC encapsulation which conveys important information about SFC data-plane must be authenticated and/or encrypted.
- Exchanged of SFC encapsulation information must be performed via a trusted protocol.
- An adequate protocol providing the isolation of different tenants in a multi-tenant cloud network should be in place.

OTHER CHALLENGES IN SFC

- Limited end-to-end visibility because of multiple data-centers and administrative domains.
- Traffic steering in SFC can be unidirectional or bi-directional depending upon the requirement.
- SFs like IDS and stateful firewall often require traffic to be steered in both directions.

NFV SOLUTION FOR MIDDLEBOX FAILURES

Failure Cause	NFV Solution
Misconfiguration	Centralized configuration policy checking using OpenFlow protocol.
Overloading	Dynamic resource flexing and load balancing.
Physical/ Electric	Virtual Network Functions (VNFs) replacing physical devices.

SDN AND NFV BASED SERVICE FUNCTION CHAINING

SDN and NFV based SFC, SDN as an enabler of SFC

STANDARDIZATION OF SFC DELPOYMENTS

- Attempts at developing standards for SFC deployments.
- Internet Engineering Task Force (IETF) is actively developing architecture for flow classification and traffic routing between service functions.
- European Telecommunications Standards Institute (ETSI) utilizes a service function architecture based on forwarding graphs and network service header.

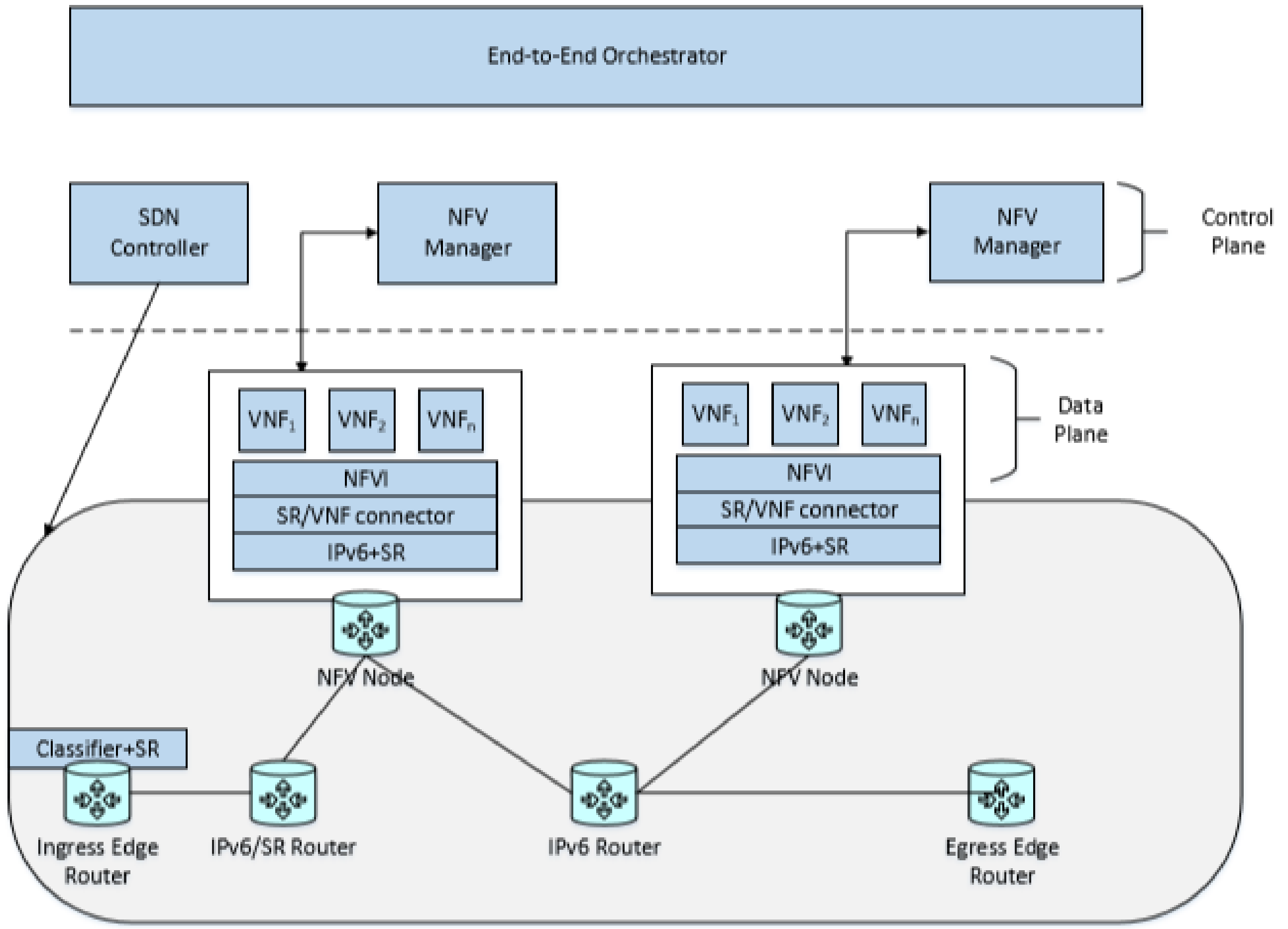
ROLE OF NFV IN SFC DEPLOYMENT

- NFV allows creation of software version of service functions such as virtual firewall, virtual router.
- NFV helps in reduction of Capital Expenditures (CAPEX) and Operational Expenditures (OPEX) by decoupling VNFs from physical devices.
- NFV can provide dynamic policy enforcement and elastic resource flexing in a network.

IPV6 SEGMENT ROUTING OVER NFVI

- Segment Routing (SR) allows node to steer a packet through a controlled set of instructions, called segments, by prepending an SR header to the packet.
- SR can be applied to the IPv6 data plane with addition of new type of Routing Extension Header.
- IPv6 based Segment Routing over Network Function Virtualization Infrastructure (NFVI) is an example of SFC over NFV.

IPV6 SEGMENT ROUTING OVER NFVI



IPV6 SEGMENT ROUTING OVER NFVI

- SRH carries a list of segments. The segment list consists of location information and traffic steering instructions.
- The original IPv6 packet is appended inside outer IPv6 packet along with SRH in the architecture.
- The edge-routers perform traffic classification and association of traffic to their respective VNF chain.
- The packet arriving at the NFV node is processed by the VNF connector, so that the packet is redirected to the appropriate VNF.

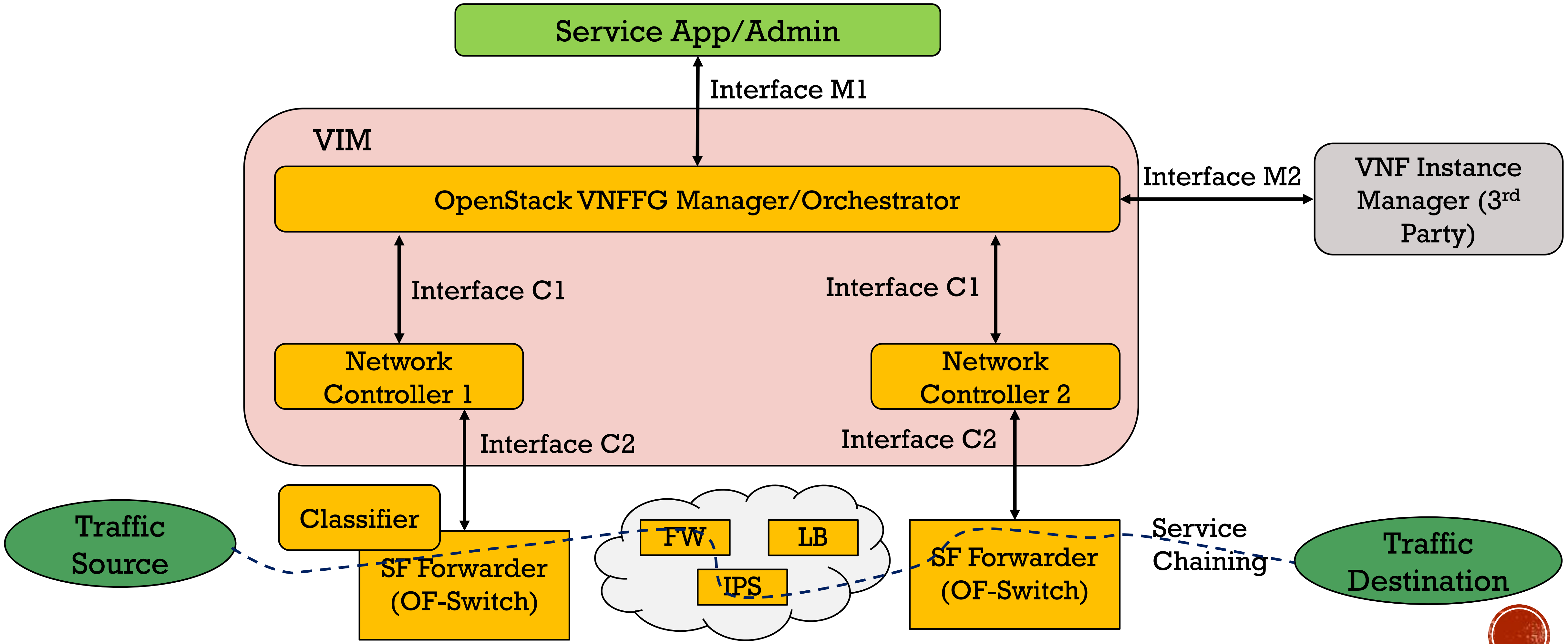
OPENNFV

- OpenNFV is a platform that facilitates the development of NFV components across various open source ecosystems.
- Focused on common configuration and deployment aspects of the end-to-end platform.
- Provides interoperability capacity between different solutions, prevents vendor lock-in, allows users to future proof their networks.

NFV VNF FORWARDING GRAPH (VNFFG)

- OpenStack based VNF forwarding graph (VNFFG) solution allows dynamic path setup between different tenants, so that flows can be steered through a sequence of VNFs.
- NFV VNFFG architecture consists of VNFFG manager, multiple OpenFlow based SDN controllers, and other components of NFVI such as traffic classifiers, service function forwarders (SFFs) hosted on vSwitch, SFs running on VMs
- Interfaces – Management Plane, Control Plane, Data Plane.

NFV VNF Forwarding Graph (VNFFG)



SDN BASED SFC

- SDN has emerged as a strong candidate to serve as a de-facto protocol for deployment of VNFs in a cloud network.
- SDN provides programming abstractions required by SFC for traffic engineering and dynamic topology control.
- SDN can dynamically manage VNF connections and underlying data-plane flows.

SDN-NFV BASED SFC COMPONENTS

Plane	Components
Management	VNFFG Manager, VNF Instance Manager
Control	SDN Controller
Data	Classifier, SF Forwarder (Proxy), VNF Nodes

VNFFG MANAGER

- Provides an interface for VNFFG service admin to specify high-level service policy intent for tenant flow from source to destination.
- The service delivery requirement should be topology agnostic.
- VNFFG translates the admin's abstract policy-based VNF service requirement into flow classification constructs as well as detailed L4-L7 VNF instance constructs.

VNF MANAGER

- Manages the lifecycle of VNFs hosted on service VMs or the external devices.
- Utilizes VNF registration API for registering the VNF instances with VNFFG manager.
- VNFFG manager imports VNF instances into VNF catalog, which is used for creation of VNFFGs.

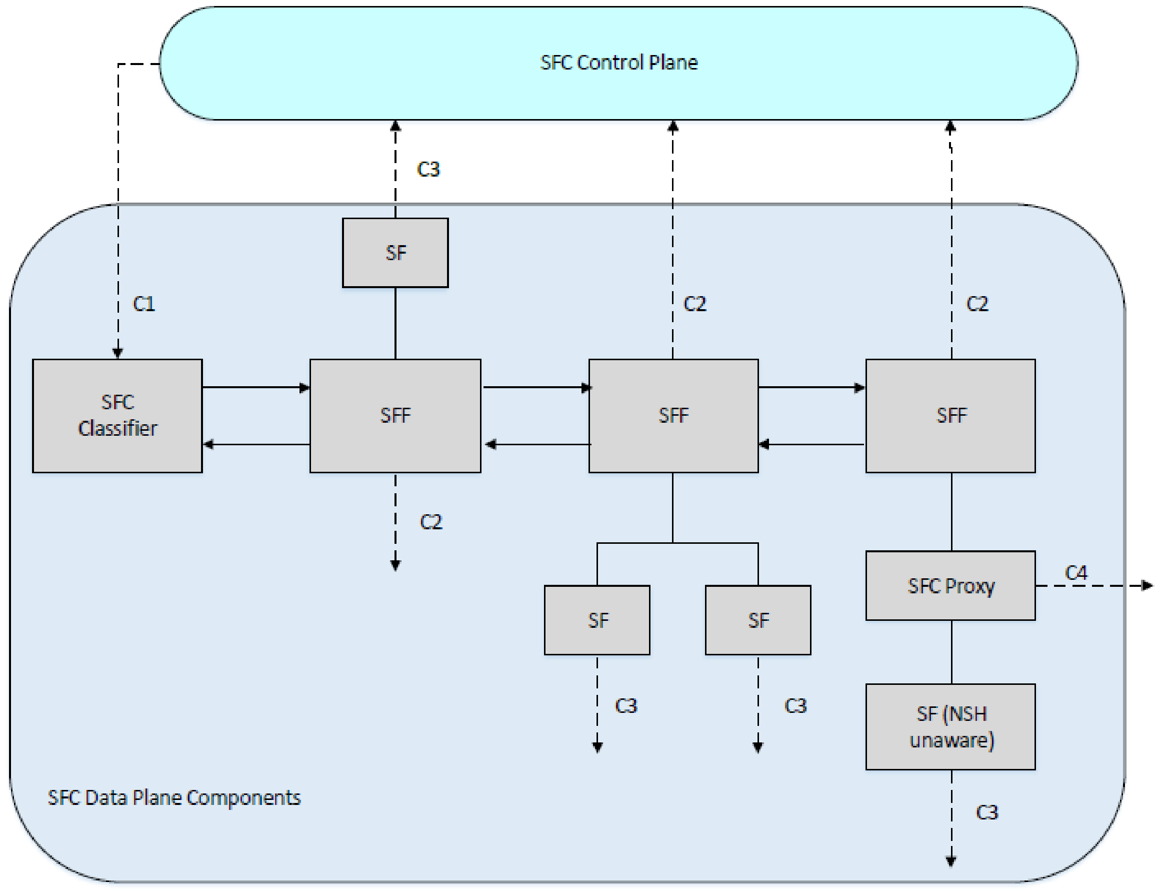
SDN CONTROLLER

- Control plane of SDN can manage SF instances, and provide mapping of SF to specific Service Function Path (SFP).
- SDN controller can install flow rules in Service Function Forwarder (SFF) devices (OpenFlow switches).
- SDN Controller manages a data-plane service domain that consists of Classifiers, Service Function Forwarders, VNFs and Proxy Devices.

SFC CLASSIFIER

- An ingress traffic classifier classifies the packet flows to different VNFFGs at the entry point of VNFFG.
- Classifier performs inspection and maps flows to VNFFGs by adding Service-Chain Header NSH encapsulation [SFC-SCH] to the packet.
- The non-SFC header aware VNFs are handled using Proxy Device.

SFC DATA PLANE COMPONENTS



SFC DATA PLANE COMPONENTS

- SFC classifier interacts with SFC control plane via interface C1.
- The SFF devices such as Open vSwitch (OVS) report connectivity status of SFs attached to them to the control plane (interface C2) via OpenFlow protocol.
- Interface C3 is used by control plane to collect packet statistics and metadata from the NSH aware SFs.
- The NSH unaware SFs make use of SFC proxy over interface C4.

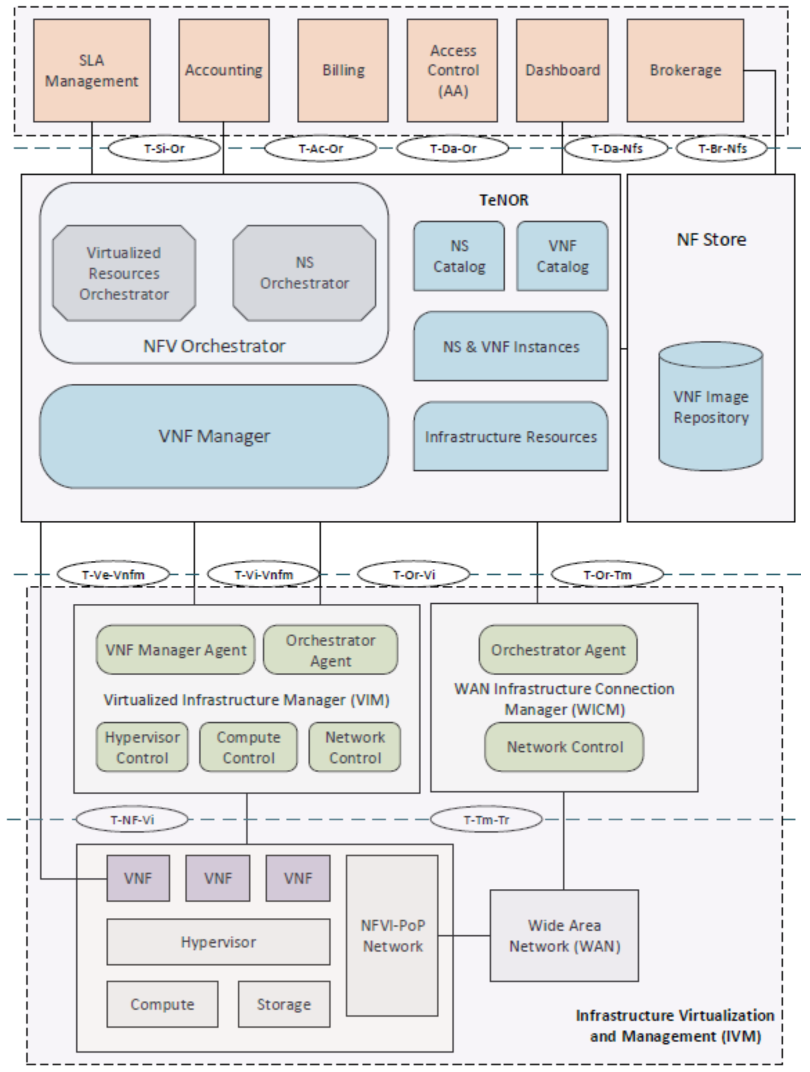
SERVICE FUNCTION CHAINING DEPLOYMENTS

T-Nova: SDN-NFV based SFC Framework, Tacker: OpenStack based SFC deployment

T-NOVA: SDN-NFV BASED SFC

- Existing Management and Network Orchestration (MANO) deployments are based on the static configuration for the nodes with VNFs.
- Inconsistencies between the workload requirements and features offered by the VNF node.
- Absence of monitoring capability that covers not only Network Function Virtualization Infrastructure (NFVI) but also VNF services.

T-NOVA: SDN-NFV BASED SFC



T-NOVA: SDN-NFV BASED SFC

- T-Nova automates the resource discovery, service mapping, service deployment and monitoring.
- T-Nova showcases the performance gains obtained using hardware acceleration.
- Based on European Telecommunication Standard Institute (ETSI) NFV ISG model.

T-NOVA: SDN-NFV BASED SFC

- SDN based Management and Network Orchestrator (MANO) framework.
- T-Nova is built using OpenDaylight SDN controller and OpenStack cloud as key technologies.

T-NOVA SERVICE ORIENTED ARCHITECTURE

- Network Functions Virtualization Infrastructure (NFVI) layer consists of network services, physical and virtual layers (switches, routers, VMs).
- NFV Management layer, composed on Virtualized Infrastructure Manager (VIM) and Wan Infrastructure Connection Manager (WICM).

T-NOVA SERVICE ORIENTED ARCHITECTURE

- TeNOR is the orchestration layer for T-Nova. TeNOR is a store of all published VNFs.
- The design goal for TeNOR is lifecycle management of distributed and virtualized NFVI.

NETwork FLOws for Cloud (Netoc)

- Marketplace layer consists of all business functions and customer-facing interfaces, which allow external users to interact with T-Nova system.

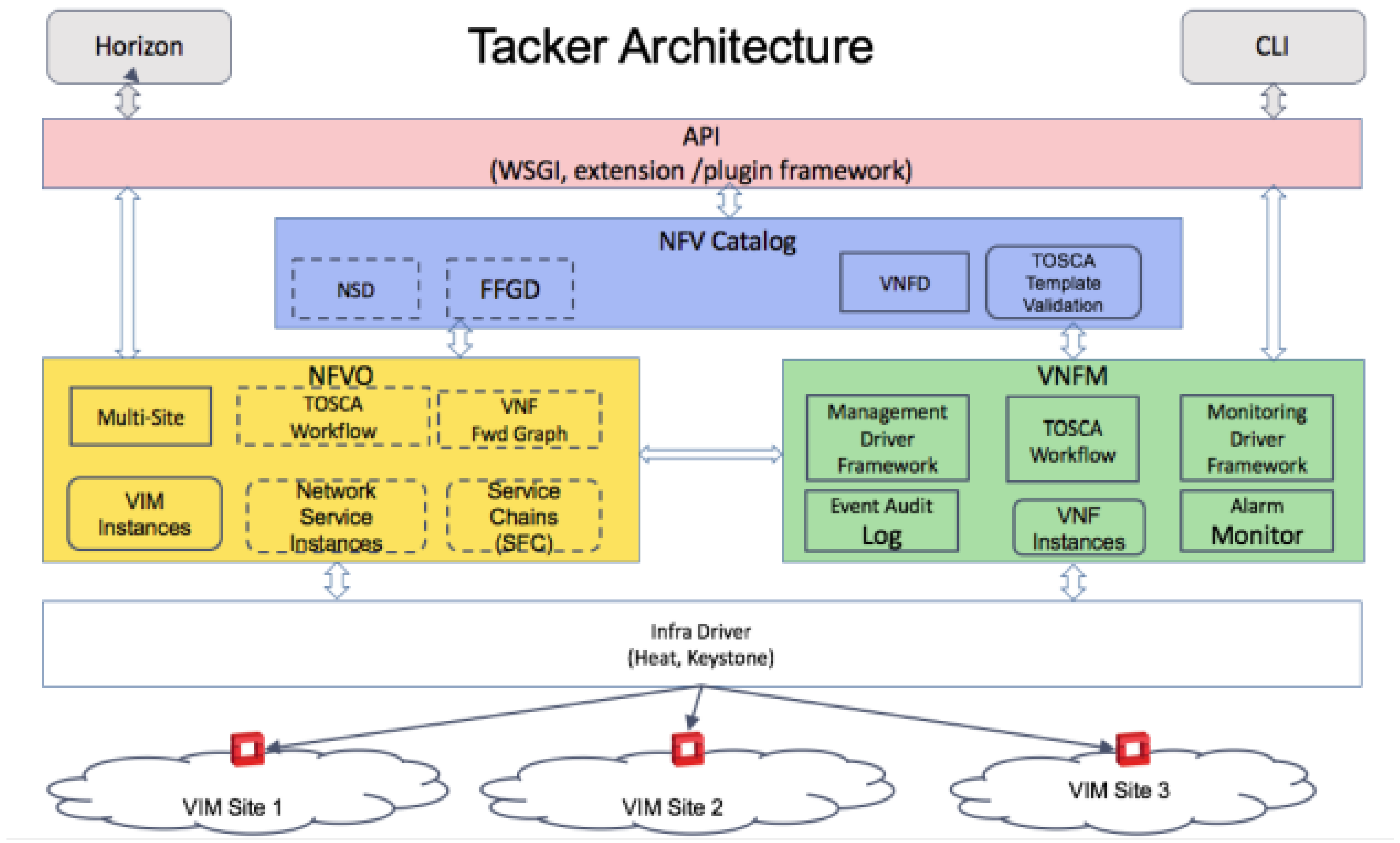
NETWORK FLOWS FOR CLOUD (NETFLOC)

- Opensource SDK for data center network programming development.
- NetFloc has integration support for SDN controller OpenDaylight.
- NetFloc provides Java interfaces and REST-based APIs for network programmability.
- End-to-end OpenStack-based data-centric network with OpenFlow enabled switches for leveraging SDN-based MANO.

SDK4SDN

- Deployed in Lithium release of the OpenDaylight controller.
- Several important use case applications such as tenant isolation using lightweight non-GRE/VxLAN tunneling mechanism.
- Layer 2 resilience using redundant datapath abstraction for link recovery.
- Virtual switch reconfiguration and SFC for providing VNF traffic steering support.

TACKER: OPENSTACK BASED SFC



TACKER: OPENSTACK BASED SFC

- OpenStack-based VNF MANO platform.
- ETSI MANO architecture and provides a functional stack to orchestrate end-to-end network services across VNFs.
- System consists of two parts, one is tacker system and another is VIM systems

TACKER COMPONENTS

- NFV Catalog consists of VNF Descriptors, Network Service Descriptors and VNF Forwarding Graph Descriptors.
- VNFM: VNF Manager is responsible for basic lifecycle management operations such as create/ update/delete, platform aware NFV load optimization, and health monitoring.
- NFVO: NFV Orchestrator is responsible for VIM resource check and allocation, SFC management.

CURRENT SFC DEPLOYMENT MODELS

SFC Solutions	OpenDaylight	OpenStack	ONOS	OpenContrail	Netfloc
NSH Spec	✓	✓	✓	✗	✗
OVS	✓	✓	✓	vRouter	✓
Plugin	✓	✓	✓	✓	✓
Encapsulation	VxLAN/GRE	MPLS	✓	MPLS over GRE/UDP & VxLAN	✗
Match	NSH-based	Neutron ports	OpenStack-based	N-tuple match packets	Neutron-ports

POLICY AWARE SERVICE FUNCTION CHAINING

PGA: Graph-based Policy Expression and Reconciliation, Policy Composition Example, Group-based Policy

POLICY AWARE SFC

- Security and traffic steering policies dictate how the traffic traverses between two endpoints in a cloud network.
- Policies help ensure optimal performance, redundancy, authentication and data integrity.
- Ensuring fulfillment of these objectives in SFC is a complex issue.

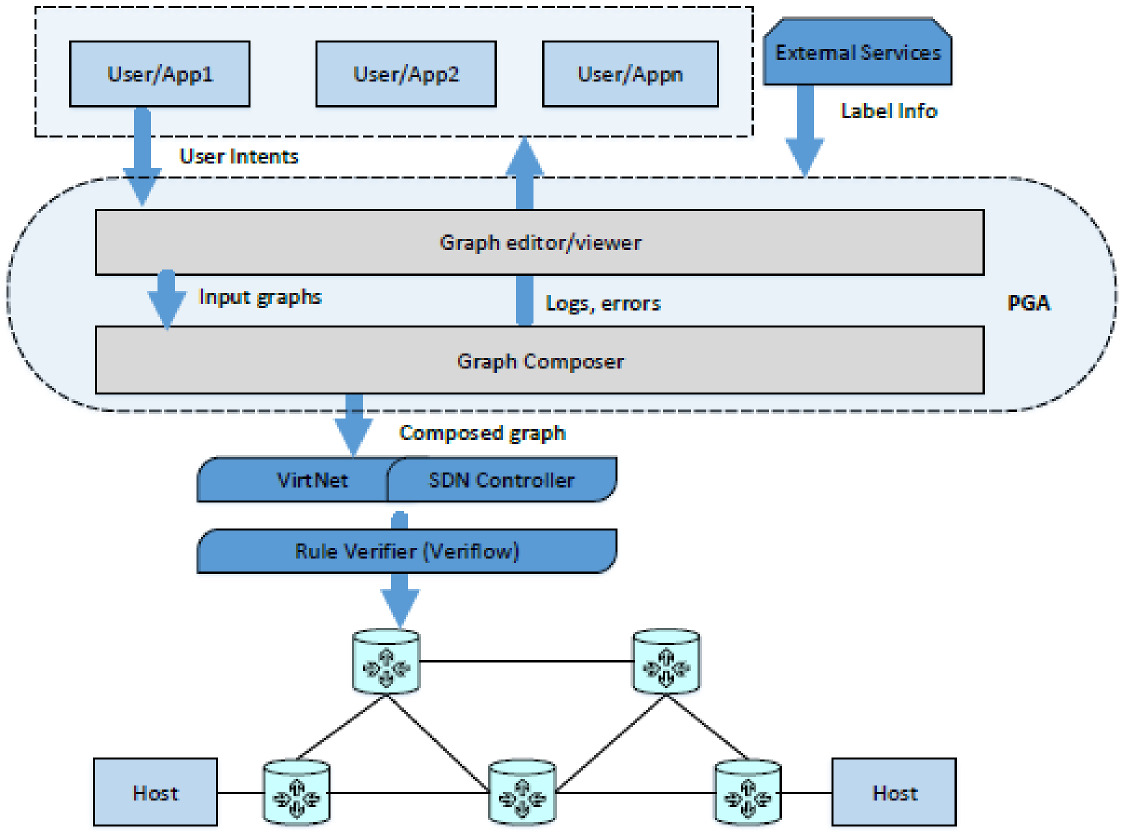
POLICY AWARE SFC

- Several Actors: Application Service Provider (ASP) , Internet Service Provider (ISP), Telecom Service Provider (TSP).
- Policies can be static or dynamic (created on the fly).
- Efficient policy formulation and compliance in the context of SFC is essential.

POLICY AWARE SFC REQUIREMENTS

- Each policy writer should be able to specify service chain policies independently.
- SFC should allow eager policy composition, satisfying the composition intent of the individual policies.
- The framework must be automated, and free of any ambiguities for the network traffic.

PGA: Graph-Based Policy Expression and Reconciliation



PGA: Graph-Based Policy Expression and Reconciliation

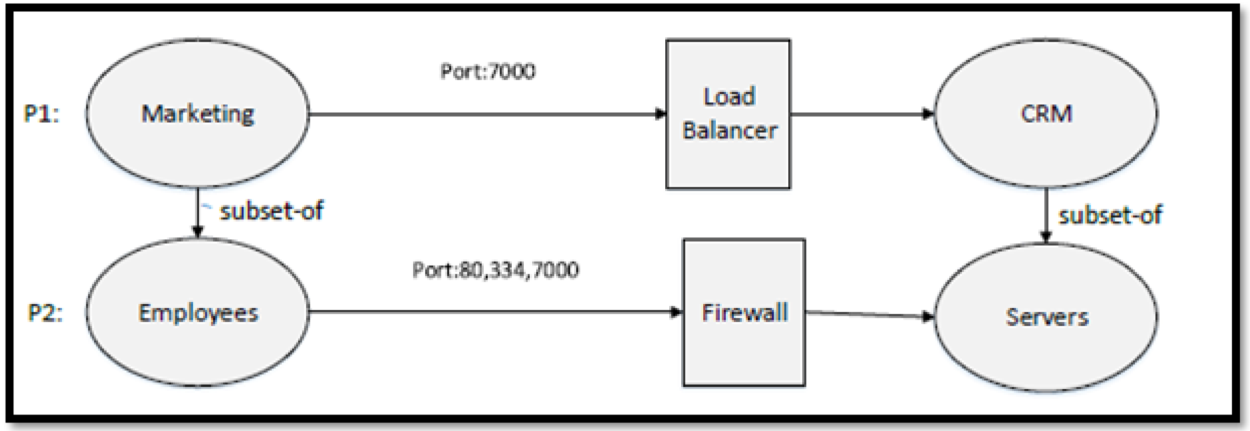
- Natural expression and automatic reconciliation of policies in different application scenarios such as tenant network and enterprise network is a difficult task.
- Coordination between network admin, firewall admin, application security consultants manual and error prone.
- Policy Graph Abstraction (PGA) provides fast, conflict-free composition of network policies.

PGA: GRAPH-BASED POLICY EXPRESSION AND RECONCILIATION

- PGA expresses network policies as graph structure.
- User/admins/tenants can independently compose policies and submit through Graph Compose UI.
- A policy specified at high-level can be composed to low-level network configuration rules.

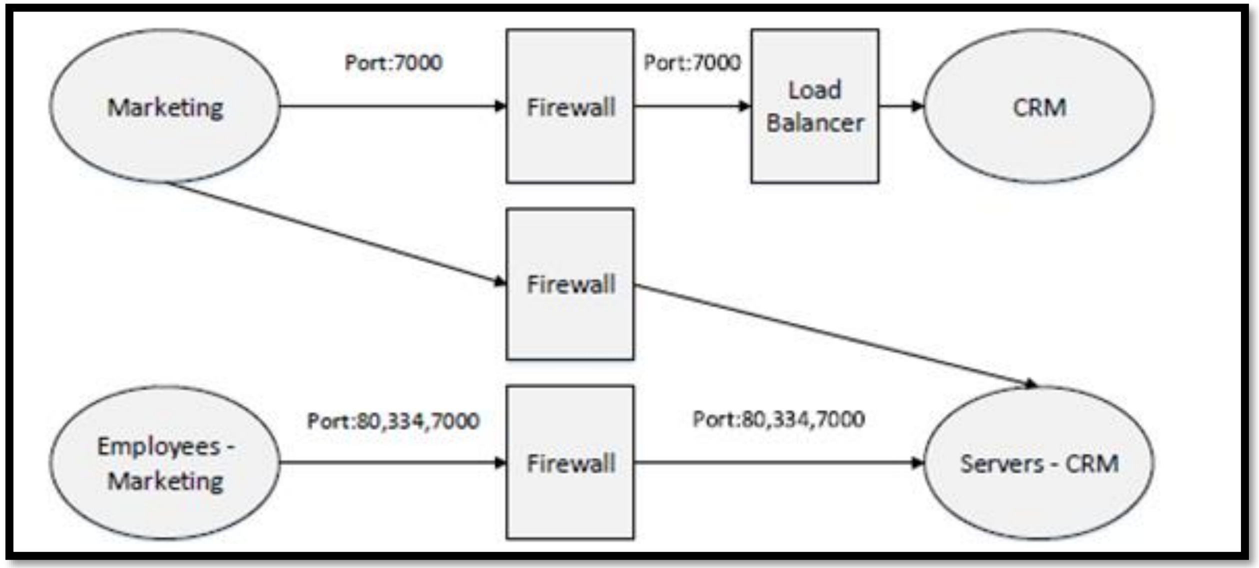
PGA: POLICY COMPOSITION EXAMPLE

- P1: Only marketing employees can send traffic to CRM server over port 7000.
- P1: The traffic should go through load-balancer.
- P2: All employees should access the installed servers through ports 80,3000,7000.
- Issue: $P1 \subseteq P2$. The non-marketing employees will be able to send traffic to CRM server.

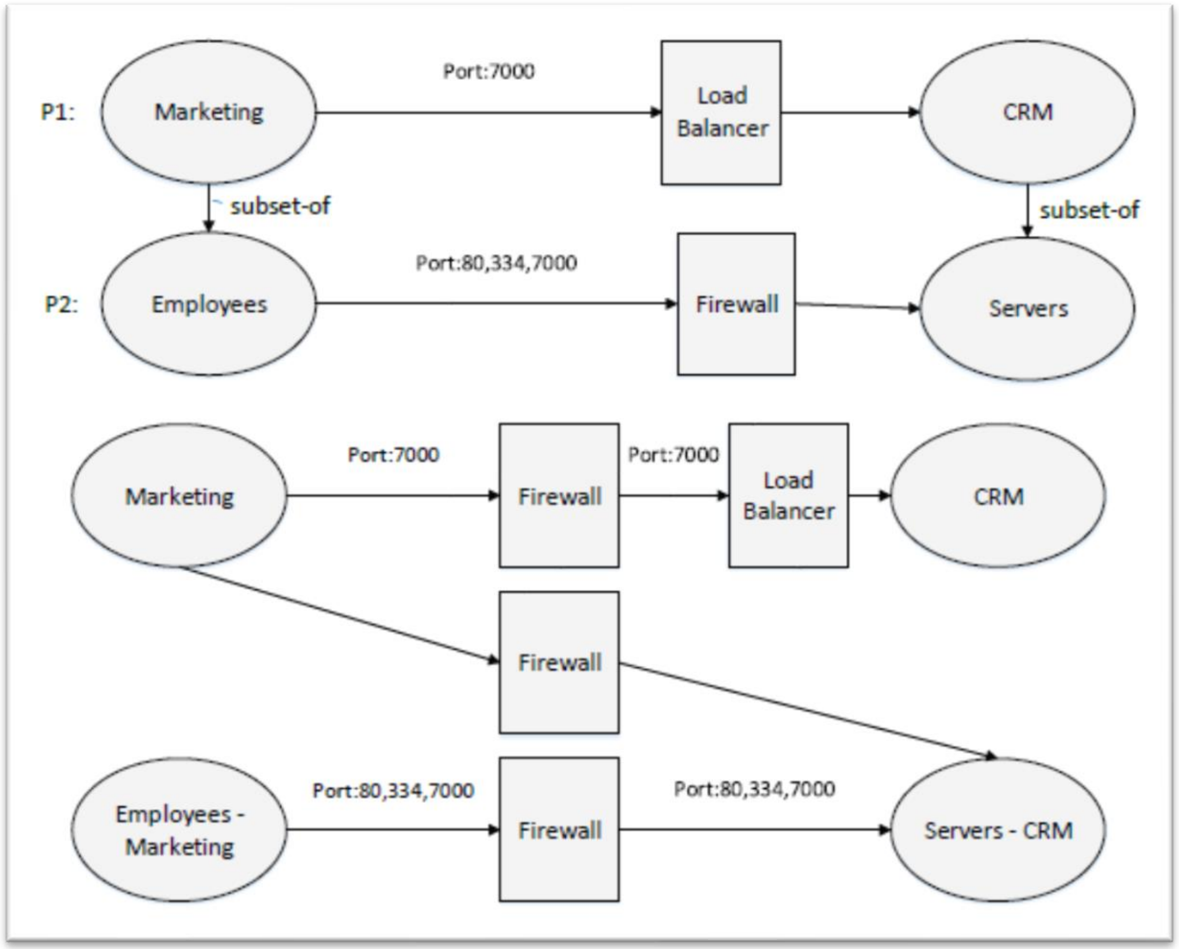


PGA BASED RESOLUTION MECHANISM

- The Marketing employees will need to go through Firewall for accessing any server which is not CRM.
- All employees who are not part of marketing will need to go through Firewall SF.
- Achieves the desired service delivery objectives but
- Can lead to performance overhead.



PGA: Policy Composition Example



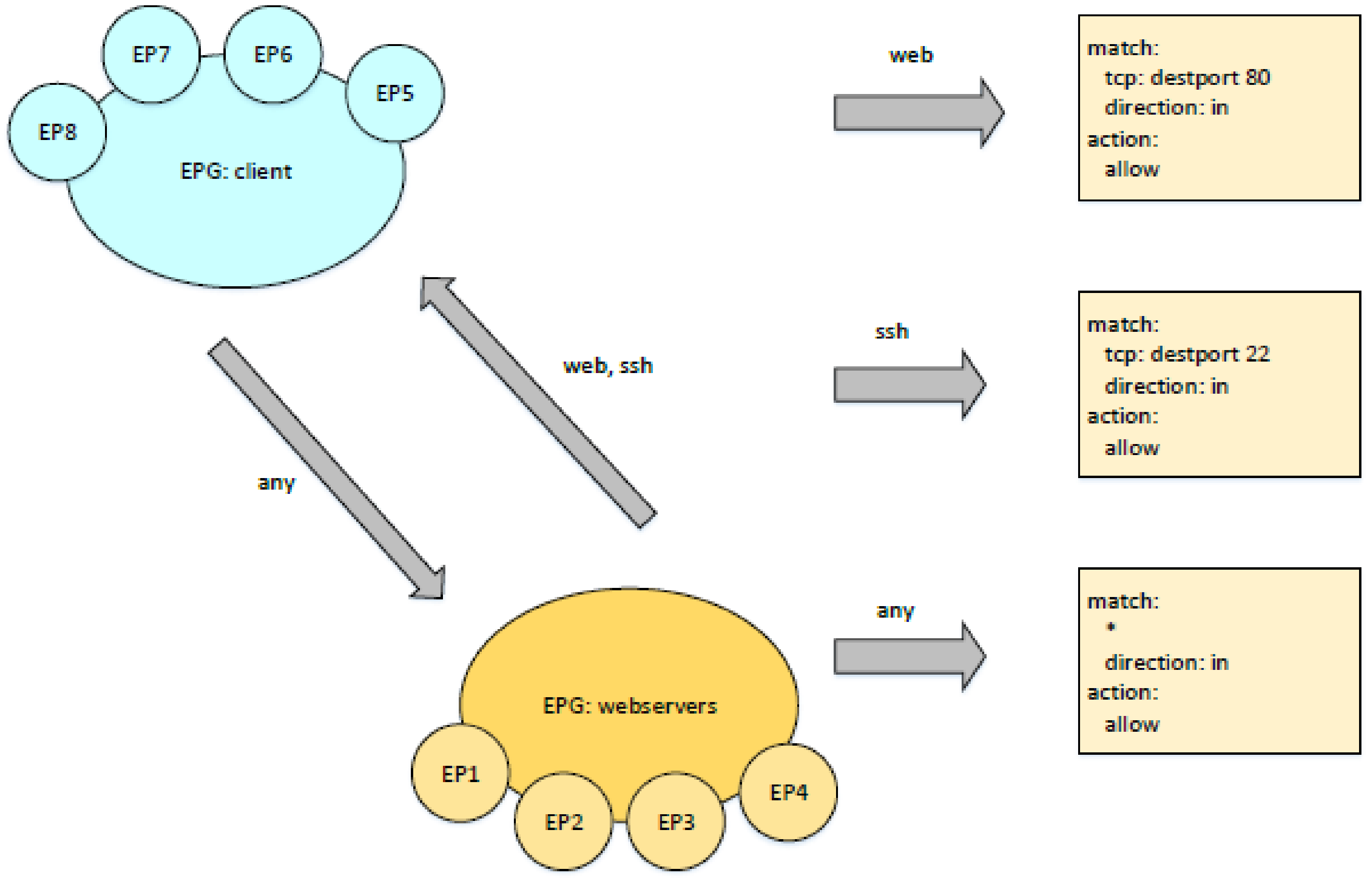
GROUP BASED POLICY (GBP)

- Automated intent driven mechanism for mapping the subscriber traffic to the service functions (SFs).
- Mapping between the intent expressed and renderers, that provide the capability to satisfy the intent.
- Endpoint: The entity that wants to communicate, e.g., Container, Network Port.
- Endpoint Group (EPG): The endpoints that consist of common attributes.

GROUP BASED POLICY (GBP)

- Contracts: Determines the scope of communication for the endpoints.
- EPGs can provide or consume contracts.
- The requirements and capabilities of EPGs are determined by the intent expressed.

GROUP BASED POLICY EXAMPLE



```
match:
  tcp: destport 80
  direction: in
action:
  allow
```

```
match:
  tcp: destport 22
  direction: in
action:
  allow
```

```
match:
  *
  direction: in
action:
  allow
```

GROUP BASED POLICY EXAMPLE

- EPG: Webservers are providing the contracts ssh and web.
- EPG: Client can subscribe to the services provided as part of the web, ssh contracts.
- EPG: Client is also providing the contract 'any' that is subscribed by EPG: Webservers.
- Traffic with destination ports 22, 80 (web, ssh) is allowed from EPG: WebServer to EPG: Client, whereas for the opposite direction all traffic is allowed.

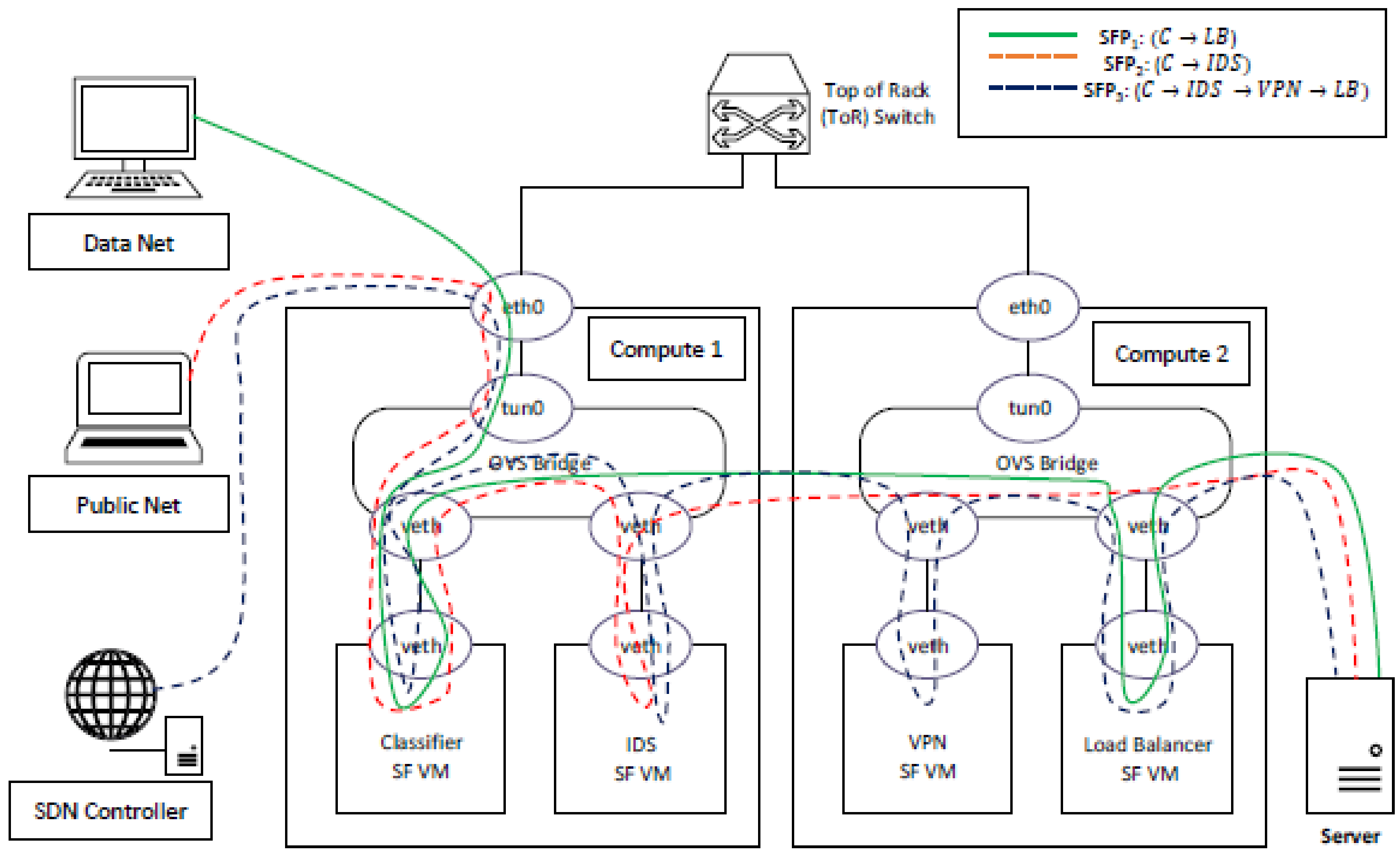
SECURE SERVICE FUNCTION CHAINING

Secure In Cloud Chaining, SFC using Network Security Defense Patterns

CLOUD SFC REQUIREMENTS

1. Traffic coming into the network should be classified into different categories based on source IP address using Classier SF.
2. Any traffic not part of data network security domain should be processed via Intrusion Detection System (IDS).
3. Data network traffic and SDN controller traffic should go through Load-balancing SF.
4. Control plane traffic from SDN controller should be encrypted using public key encryption scheme.

CLOUD SFC EXAMPLE



CLOUD SFC EXAMPLE

- OVS bridges can communicate with each other using a tunnel network tun0.
- SDN controller can communicate with OVS switches using the OpenFlow protocol.
- SFs are connected to OVS bridges using tap interfaces veth.
- Service Function Path (SFP) after application of security policies and operational constraints can have issues.

CLOUD SFC ISSUES

- Strategy 1 Order: C → V PN → IDS → LB.
- Issue: SDN controller traffic needs to go through both VPN and IDS as per policy.
- Placing VPN first (incorrect order) violates security objective.
- IDS should precede VPN, since VPN encrypts the traffic.

CLOUD SFC ISSUES

- Strategy 2 Order: C → LB → IDS → VPN.
- Issue: The traffic from SDN controller and data network has to go through classier and load balancer.
- Malicious traffic could have been filtered out using
- Less impact on QoS offered by the load balancer.
- Incorrect placement leads to an efficiency issue.

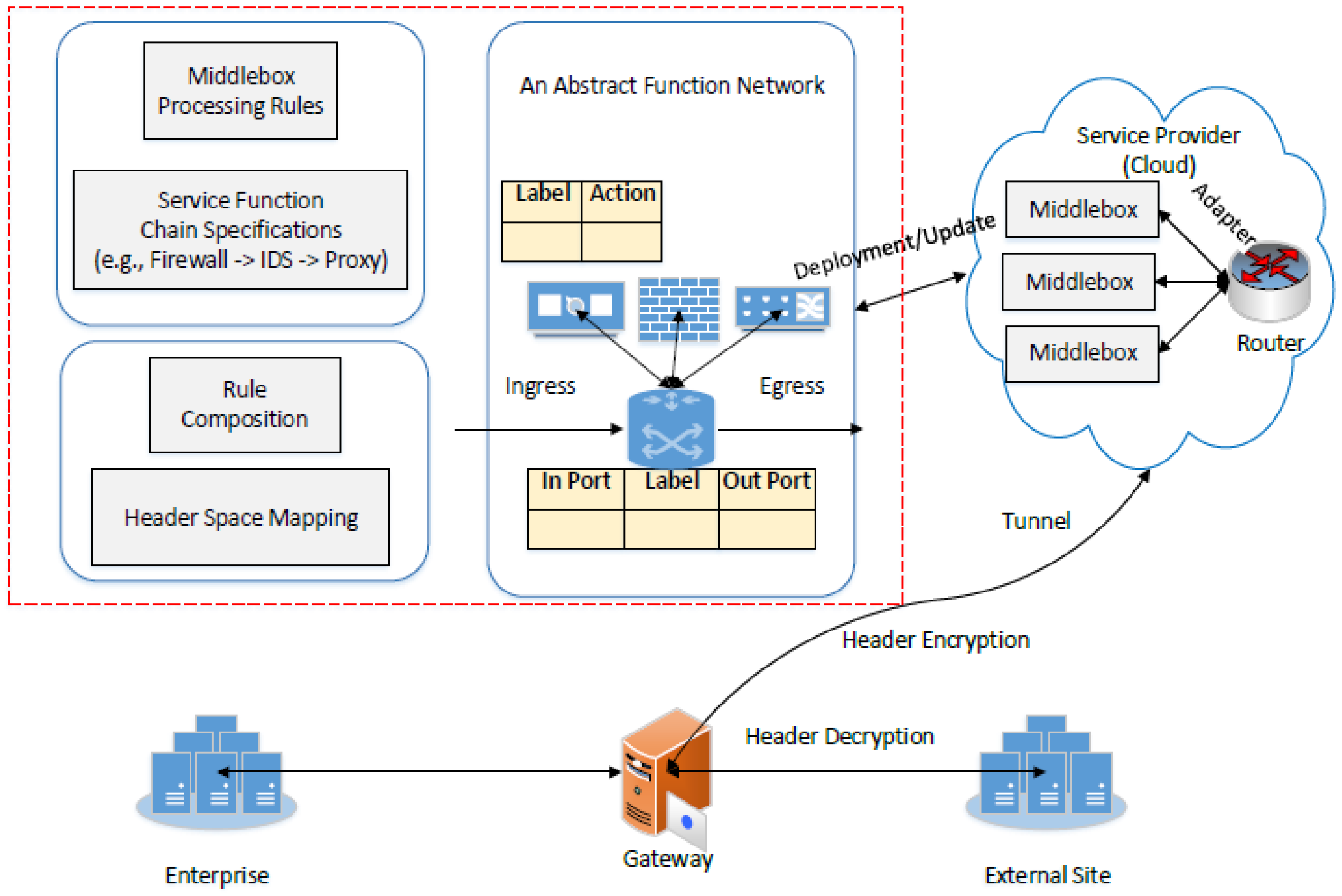
CLOUD SFC ISSUES

- Strategy 3 Order: C → IDS → VPN → LB.
- Efficient Placement: Unwanted traffic is filtered at IDS.
- Load balancer has to deal with only legitimate traffic from Data-Net and SDN Controller.
- Correct Ordering: The traffic is passed in raw (un-encrypted) format through IDS, and later through VPN.
- IDS has complete visibility.

SECURE IN-CLOUD CHAINING

- Secured out-sourcing of network functions.
- Use of 3rd party modules for providing confidentiality and integrity incurs performance overhead.
- SICS framework encrypts each packet header based on in-cloud rule matching.
- Minimal performance overhead.

SECURE IN-CLOUD CHAINING



SECURE IN-CLOUD CHAINING

- SICS gateway tunnels ingress and egress traffic to the cloud middleboxes.
- Incoming traffic headers are encrypted using AES encryption.
- Traffic is then passed through different SFs and decrypted at the enterprise gateway.
- Each packet is assigned a label to ensure appropriate SFP.

SECURE IN-CLOUD CHAINING

- The traffic is encrypted at the enterprise gateway and sent to the internet.
- The label-based approach provides efficient packet lookup.
- The lookup overhead for the matched labels using hash table is O(1).
- 16-bit label is able to represent one million 5-tuple rules.
- A label can only reveal information about the actions and forwarding behavior of the middleboxes, preserving privacy.

SFC NETWORK SECURITY DEFENSE PATTERNS (NSDP)

- Multi-tenant cloud network.
- Varying security and service provisioning requirements.
- Current VNF deployment solutions consider only the performance aspect of the SFC.
- The security needs are often loosely coupled with other SFC objectives.

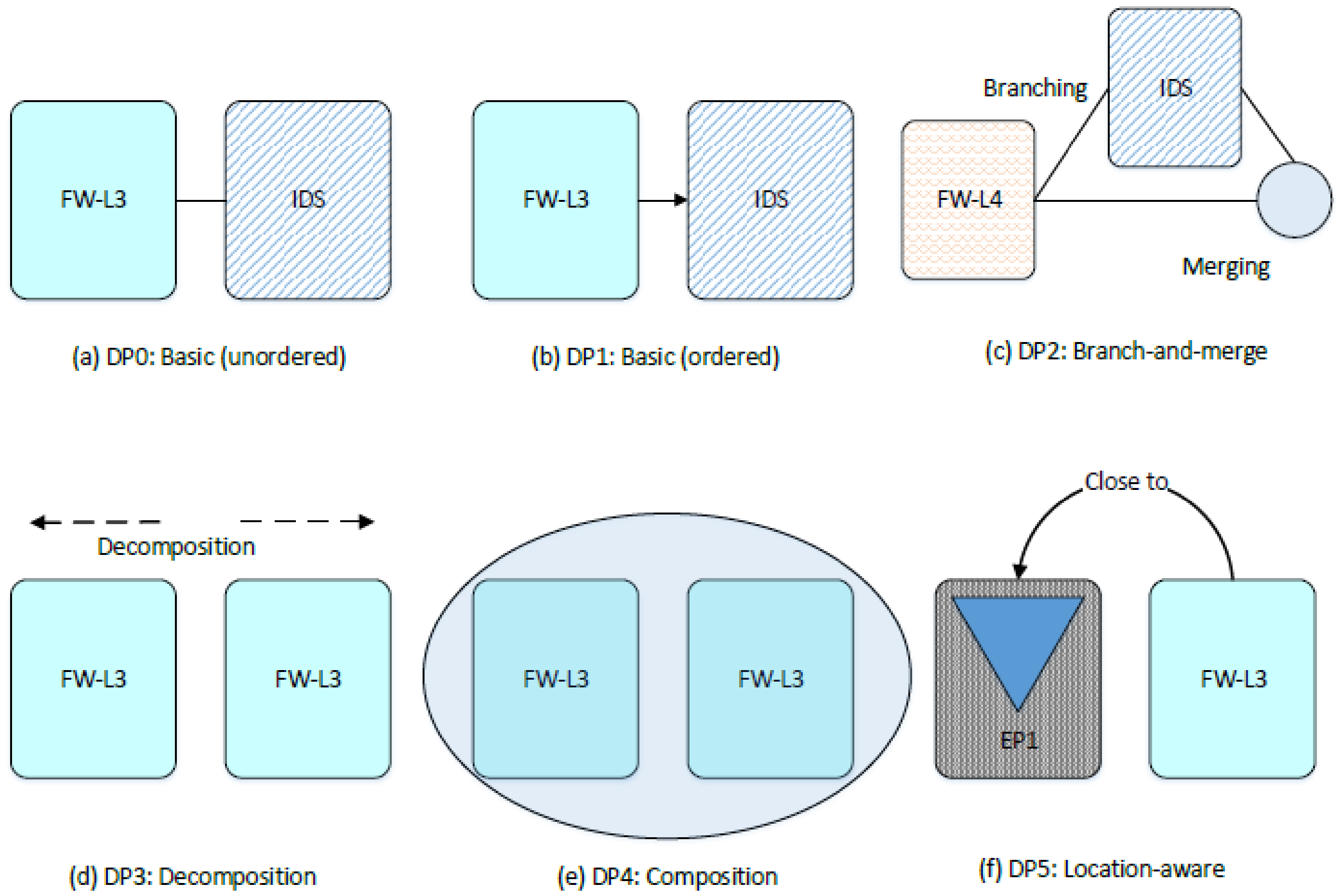
SFC NETWORK SECURITY DEFENSE PATTERNS

- High-level security needs can be mapped to security patterns.
- Security constraints can be mapped to appropriate SFC deployment constraints.
- NSDP that can help in achieving security and ordering objectives in SFC.
- Combine security patterns with deployment constraints.

NSDP DEPLOYMENT CONSTRAINTS

- Region constraint forces the SF to be placed in a specific region or zone.
- Co-Location constraint requires placement of several similar or dissimilar SFs on the same node.
- Distribution requires placement of SFs on different nodes.
- Waypoint constraint requires traversal of traffic flows from different nodes through a given SF.

SFC NETWORK SECURITY DEFENSE PATTERNS



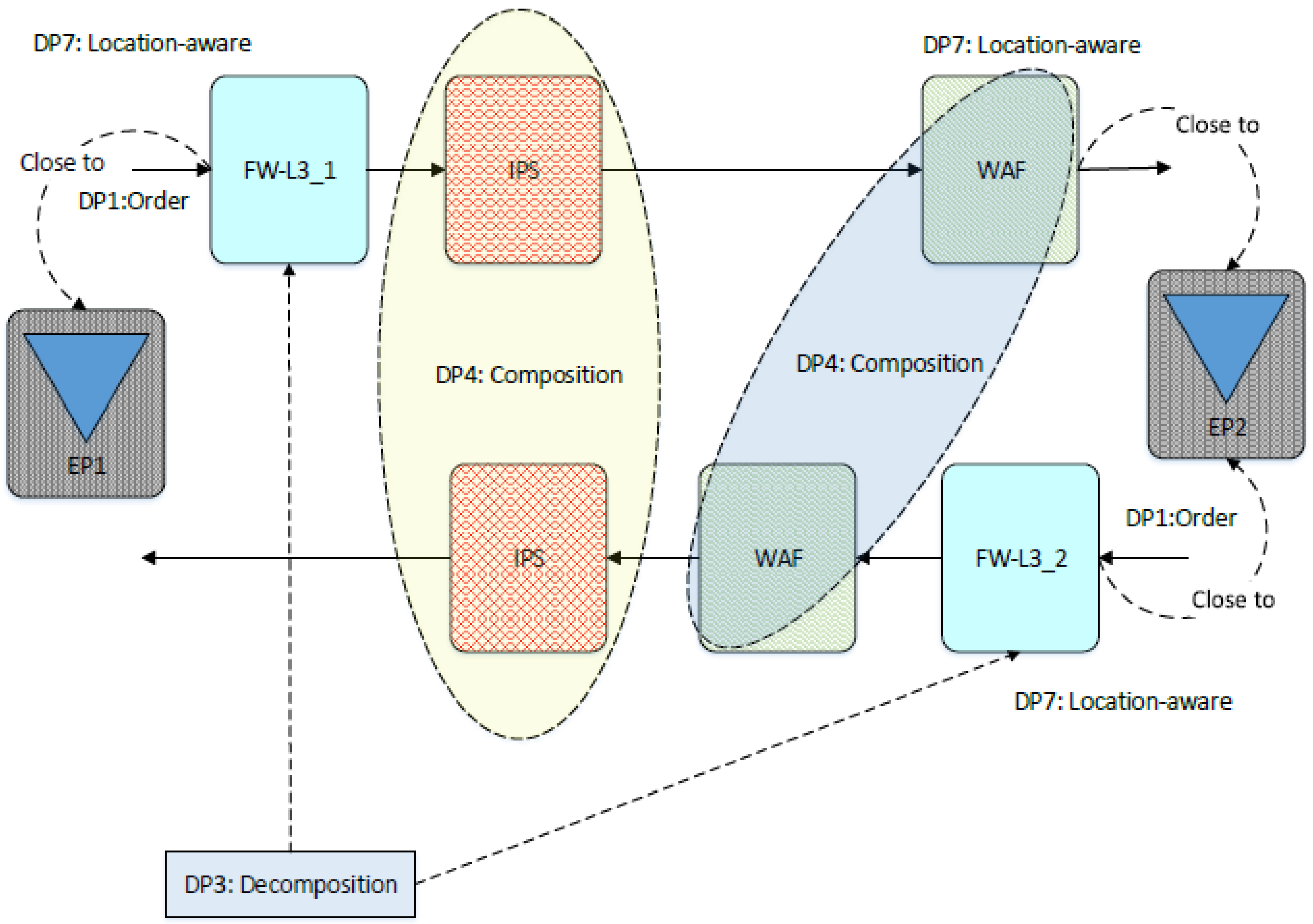
NSDP AWARE SFC

NDSP	Description
Basic Unordered	Packet traversal through SFs don't need to follow any strict ordering, e.g., L3 Firewall and IDS.
Basic Ordered	SF needs to be traversed before another in order to meet security or efficiency objectives, IDS placement before VPN.
Branch-and-Merge	Traffic is required to be split into two branches in order to pass through different service function paths.

NSDP AWARE SFC

NDSP	Description
Decomposition	SF splitting into two different subsets of the same type for the purpose of de-localization, e.g., Distributed Firewall.
Composition	Merging of the functionality of two SFs in order to achieve some desired properties such as state aware SFC, e.g., Stateful Firewall.
Probe Point	Placement of an SF at a certain distance from a location, e.g., L3 Firewall close to the endpoint and DDoS SF close to the location.

NDSP AWARE SFC



NDSP AWARE SFC

- Decomposition and Location Awareness: The Layer 3 Firewall should be decomposed and placed close to source and destination.
- Decomposition: The L3-FW is decomposed into FW L3_1 and FW L3_2.

NDSP AWARE SFC

- Location Awareness: One firewall can be placed near EP1 and another near the destination node EP2.
- Composition: We want the traffic to be steered through the IDS and Web Server.
- Combine the functionality of WAF and IDS into one SF for SFC.

CITE THIS WORK

```
@book{huang2018software,
title={Software-Defined Networking and Security: From Theory to Practice},
author={Huang, Dijiang and Chowdhary, Ankur and Pisharody, Sandeep},
year={2018},
publisher={CRC Press}}
```

REFERENCES

- https://www.sdxcentral.com/sdn/network-virtualization/definitions/what-is-network-service-chaining/
- https://tools.ietf.org/id/draft-wang-sfc-ns-use-cases-03.html#rfc.section.4.1
- https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7969147
- https://reader.elsevier.com/reader/sd/pii/S092054891730017X?token=312CBA1AF4E1D95287117794D8D000264051EF7F9B1F38B34D1C7678E6A438EA9E8E65A4FA0BC174F237415BB954440D
- https://tools.ietf.org/pdf/rfc7498.pdf
- https://tools.ietf.org/pdf/draft-ietf-6man-segment-routing-header-02.pdf
- https://www.opnfv.org/
- https://wiki.opnfv.org/display/VFG/Openstack+Based+VNF+Forwarding+Graph

REFERENCES

- https://docs.openstack.org/tacker/latest/install/index.html