

Introduction to Service Function Chaining on the OpenStack Cloud Platform

Davide Borsatti, Gianluca Davoli,
Walter Cerroni, Chiara Contoli, Franco Callegati
Department of Electrical, Electronic and IT Engineering
University of Bologna, Italy



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

6th OSM Hackfest – Santa Clara, 16th May 2019

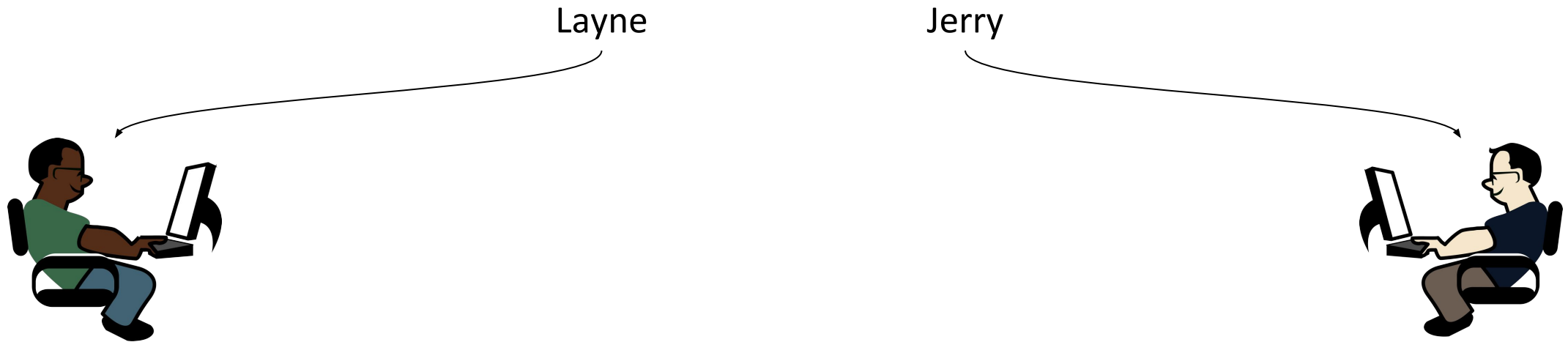
Preliminary Information

- This presentation is an extract of the one given at CNSM – SR-SFC Workshop 2018 – Rome, 5th Nov. 2018 for the paper:

D. Borsatti, G. Davoli, W. Cerroni, C. Contoli, F. Callegati, Performance of Service Function Chaining on the OpenStack Cloud Platform, Proc. of 1st Workshop on Segment Routing and Service Function Chaining (SR+SFC 2018), in conjunction with CNSM 2018, Rome, Italy, November 2018.

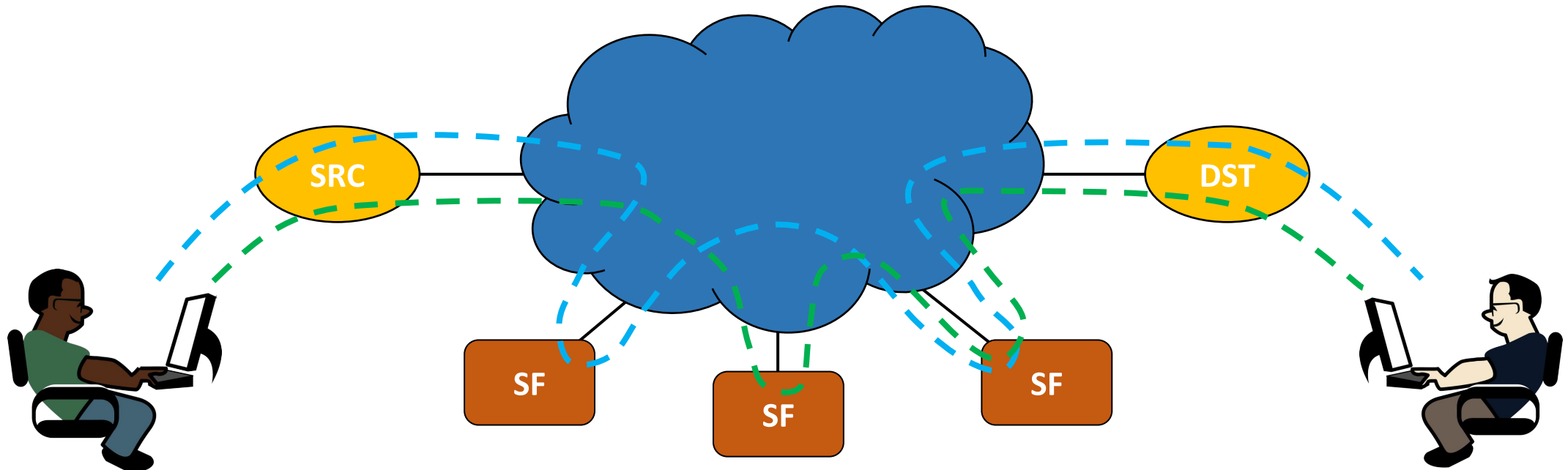
- For more informations: Davide Borsatti davide.borsatti@unibo.it
-

A couple words on Service Function Chaining (SFC)



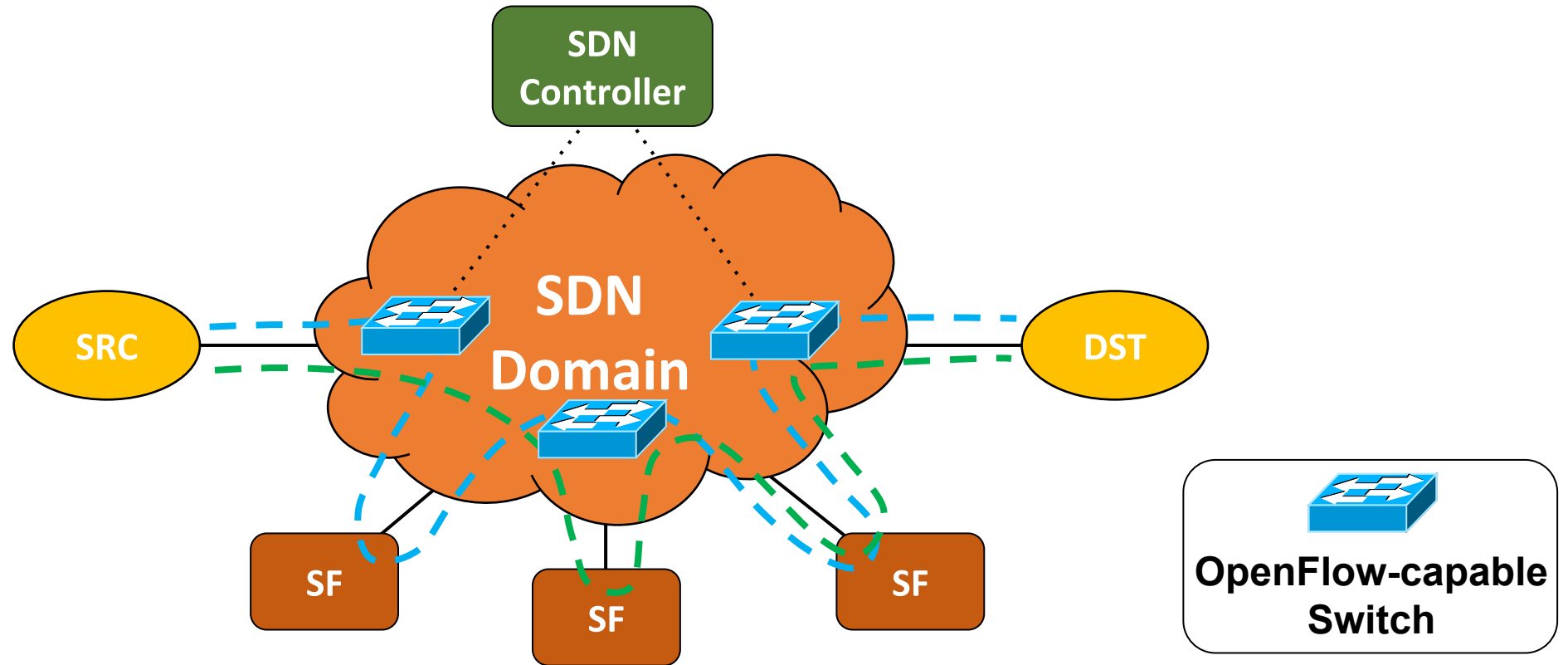
A couple words on Service Function Chaining (SFC)

Concatenation (*chaining*) of basic services or (virtualized) network/service functions...



...in order to obtain a composite service, spanning over the whole network domain

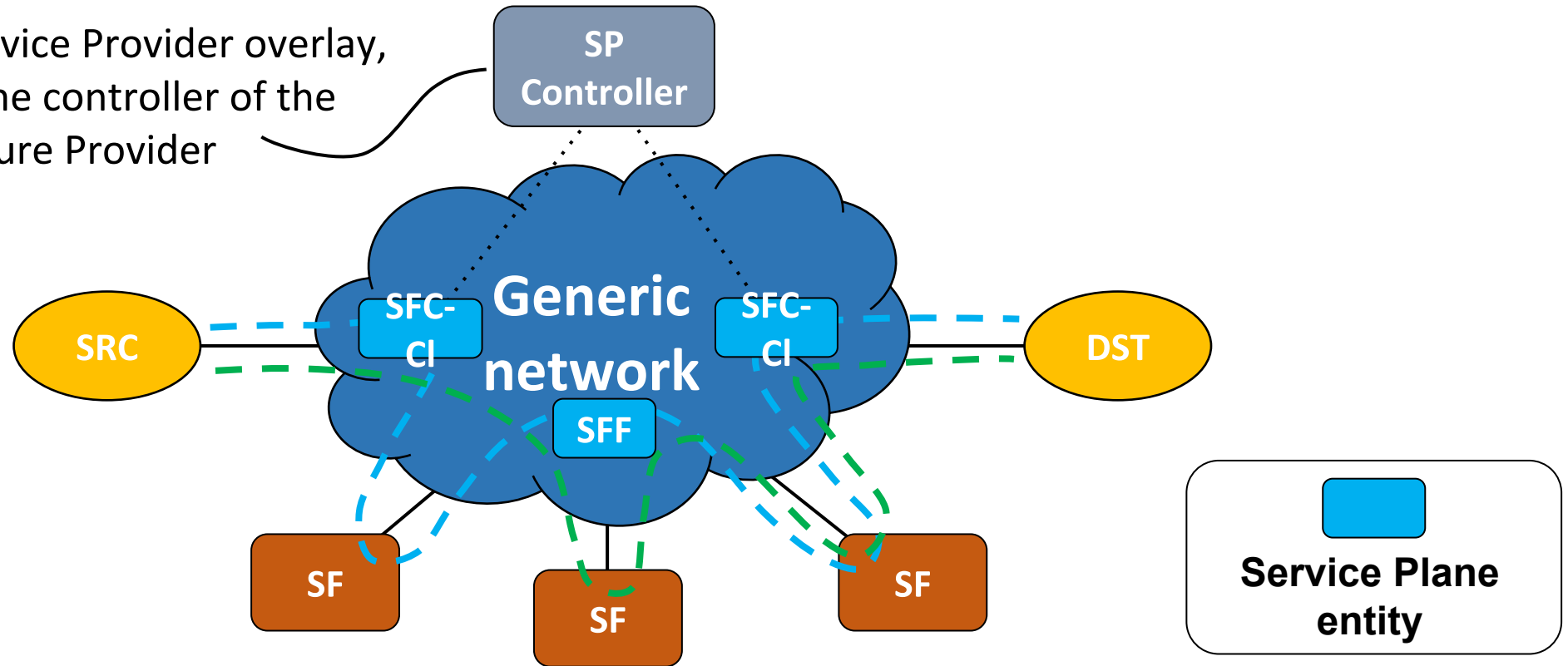
Service Function Chaining over an SDN Domain



Traffic steering is handled by OpenFlow-capable switches, controlled by a SDN Controller

Service Function Chaining over a generic network

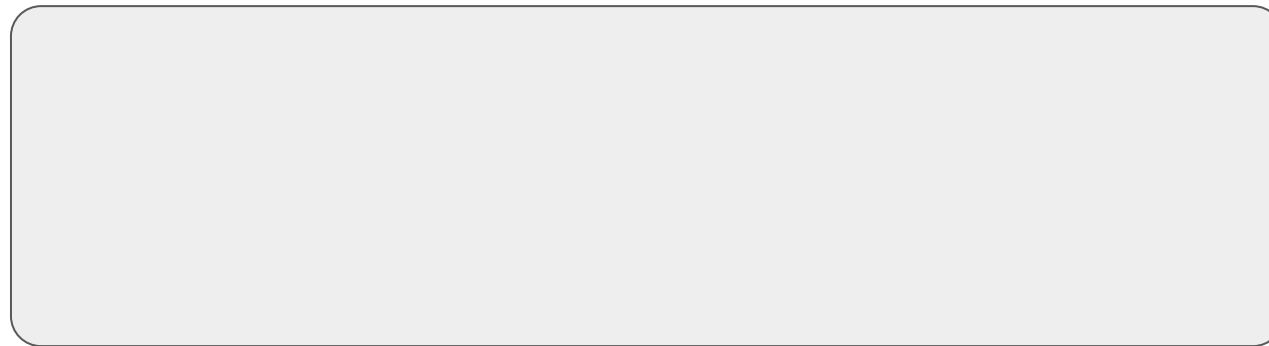
Controller of the Service Provider overlay,
independent of the controller of the
Infrastructure Provider



Traffic steering for SFC is handled by Service Plane entities [IETF, 2016]

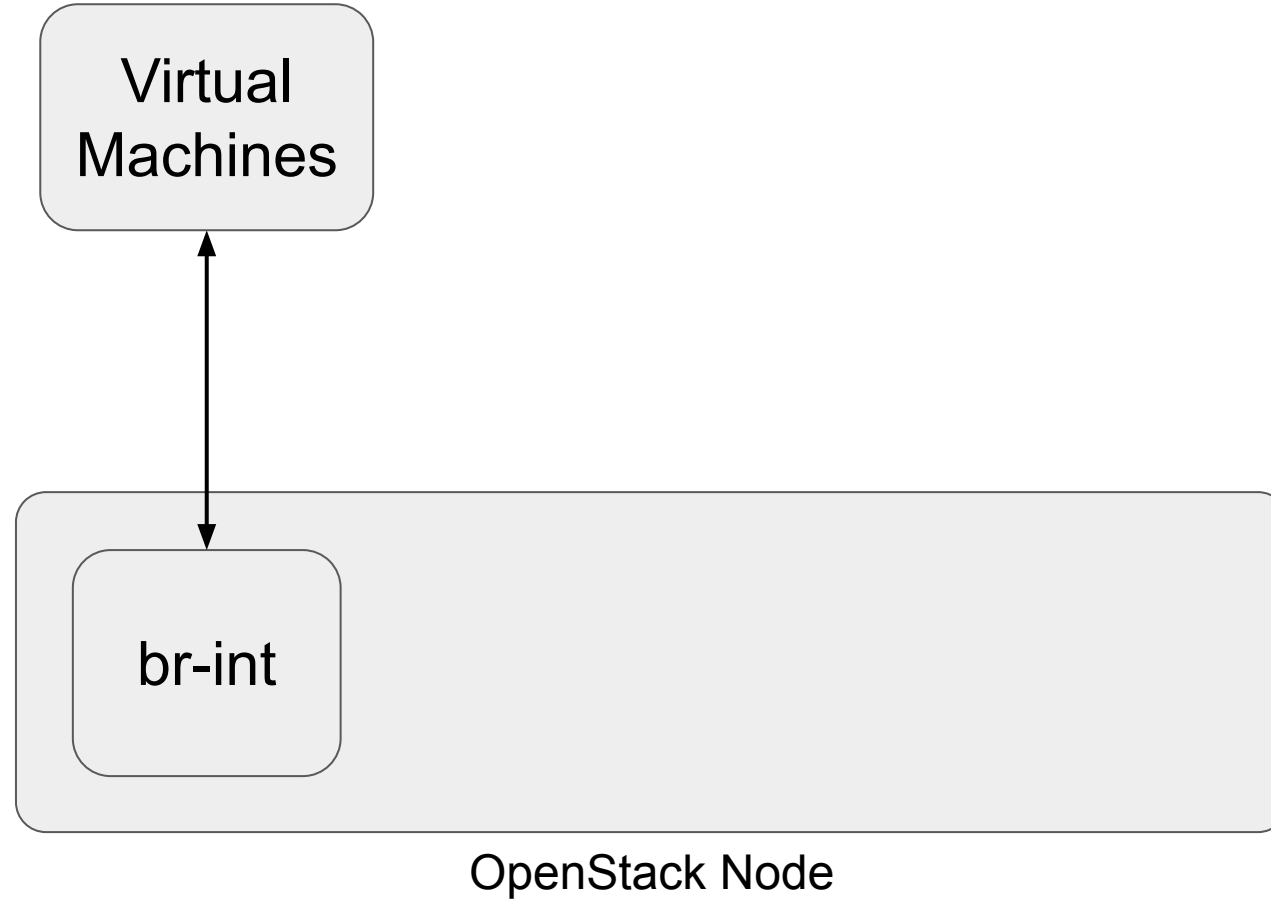
OpenStack - Neutron

It provides Network as a Service (NaaS), as the OpenStack users are able to create their own networks where to plug the Virtual Network Interface of their virtual instances. Neutron is able to separate the logical view of the network from the actual physical view, providing APIs to define, manage and connect virtual networks.

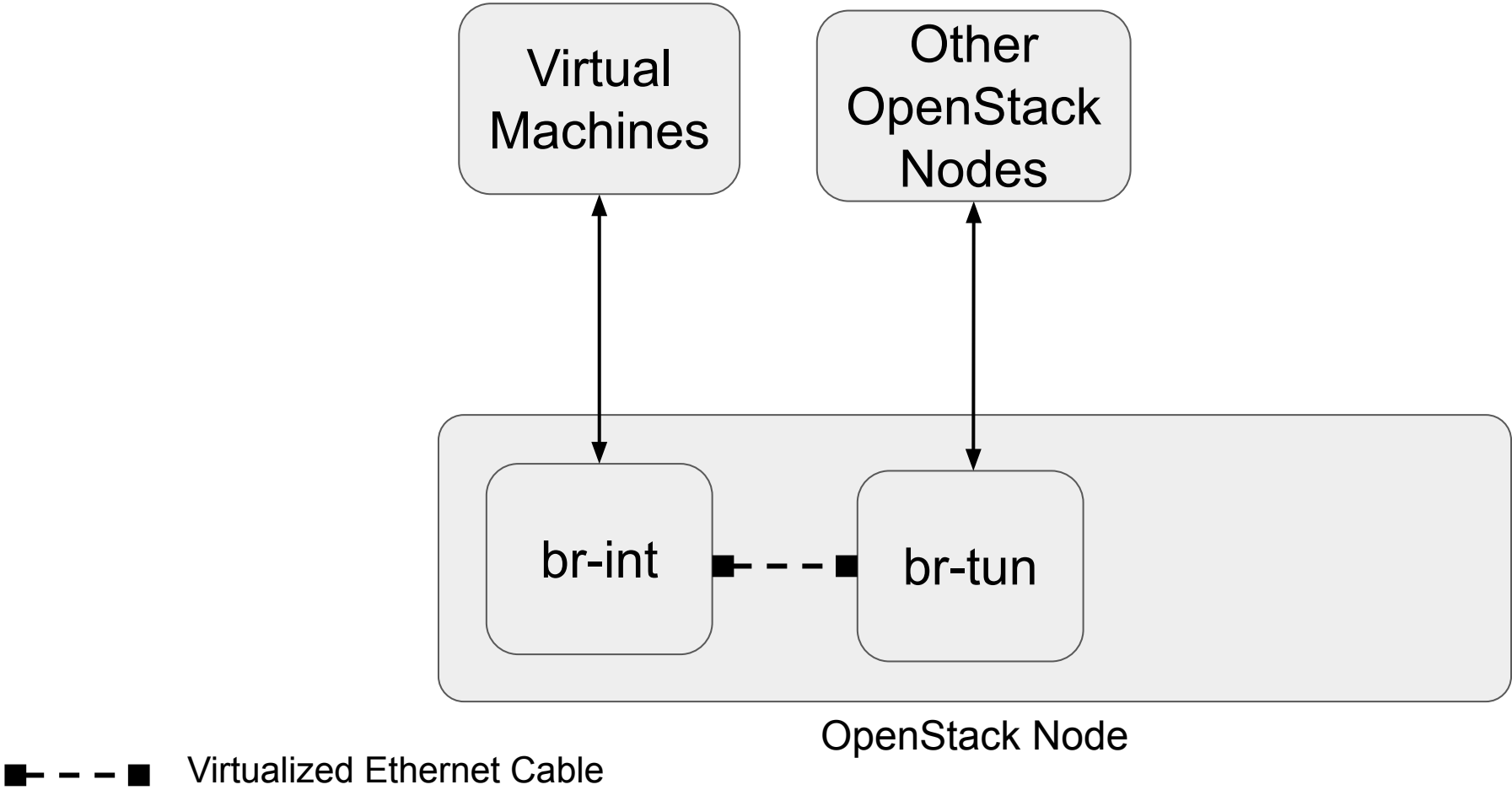


OpenStack Node

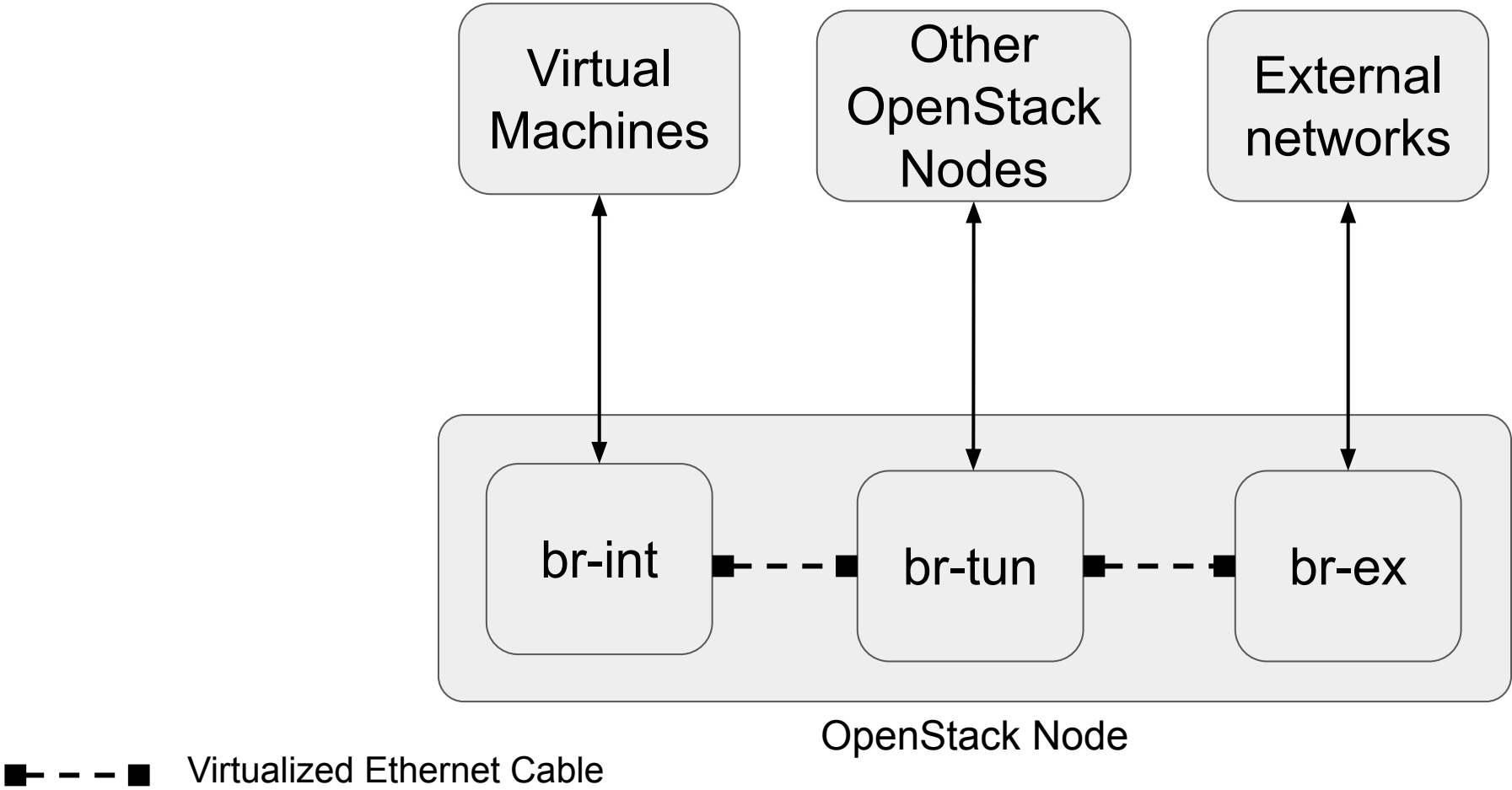
OpenStack - Neutron



OpenStack - Neutron



OpenStack - Neutron



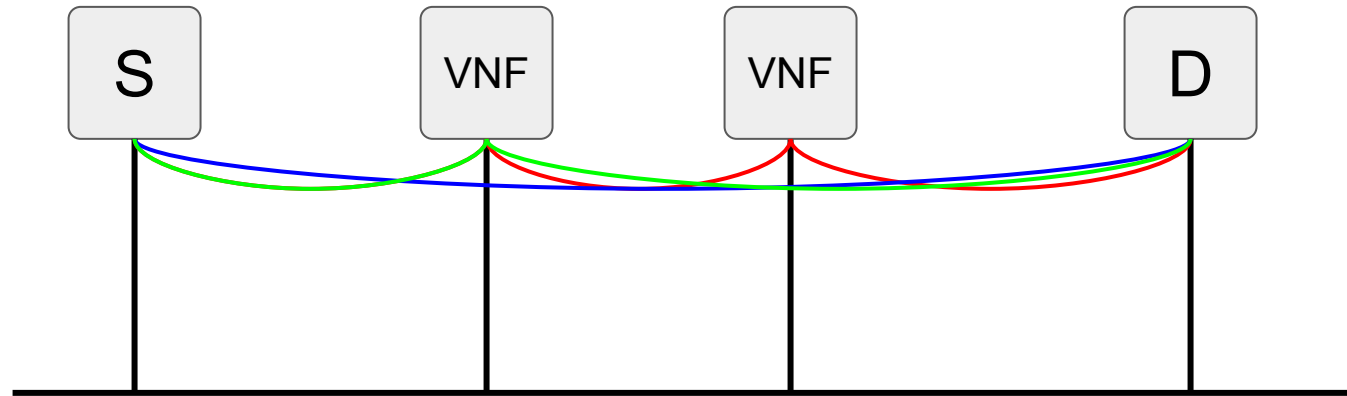
SFC-extension for OpenStack

The extension allows for the creation of SFPs, it natively supports interaction with Open vSwitch (OvS) and it implements a flow classification mechanism.

It is composed by four main entities:

1. Flow Classifier
 2. Port Pair
 3. Port Pair Group
 4. Port Chain
-

SFC-extension – Flow Classifier



Flow Classifier:



UDP traffic

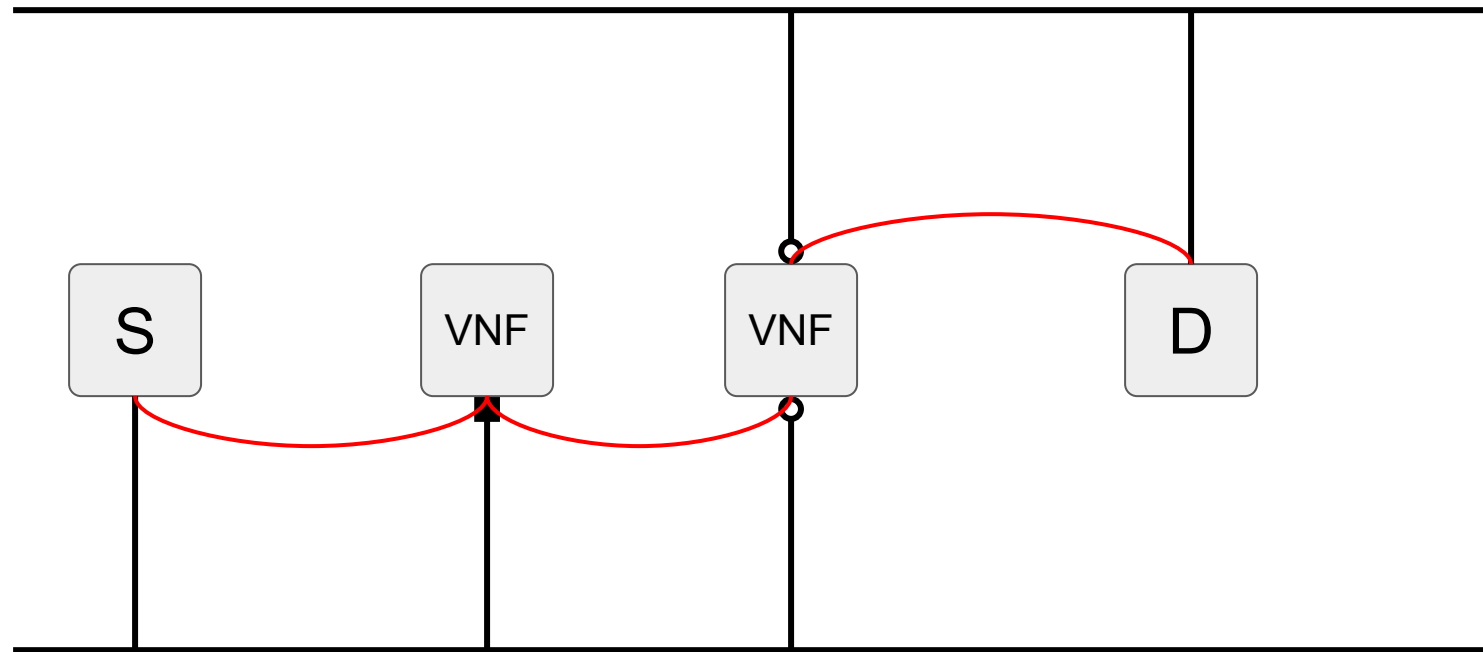


HTTP traffic (TCP port 80)



Other packets

SFC-extension – Port Pair



Port Pair:

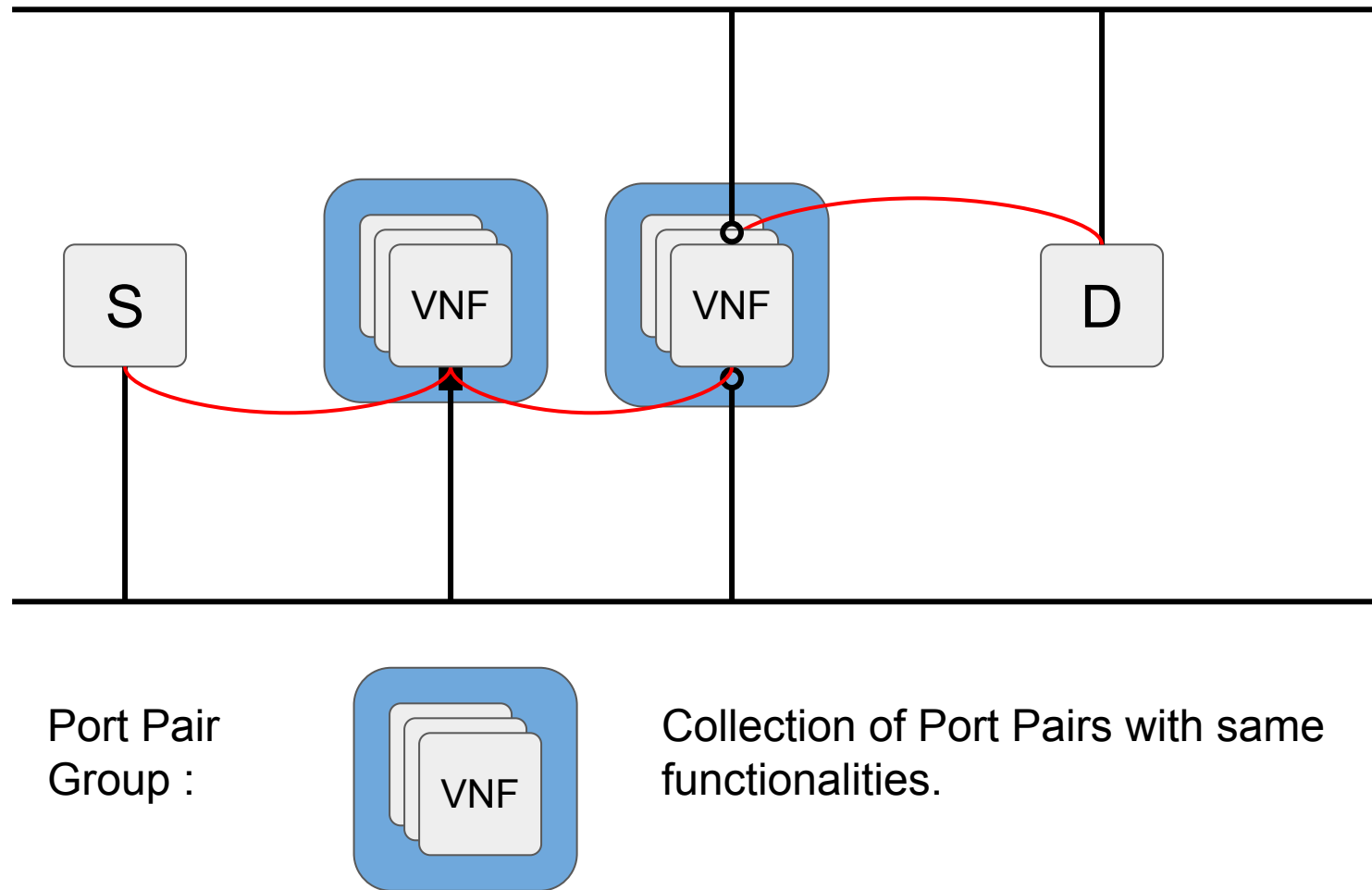


Same ingress and egress port

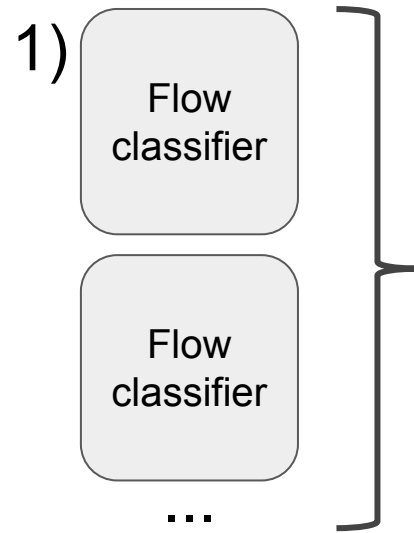


Different ingress and egress port

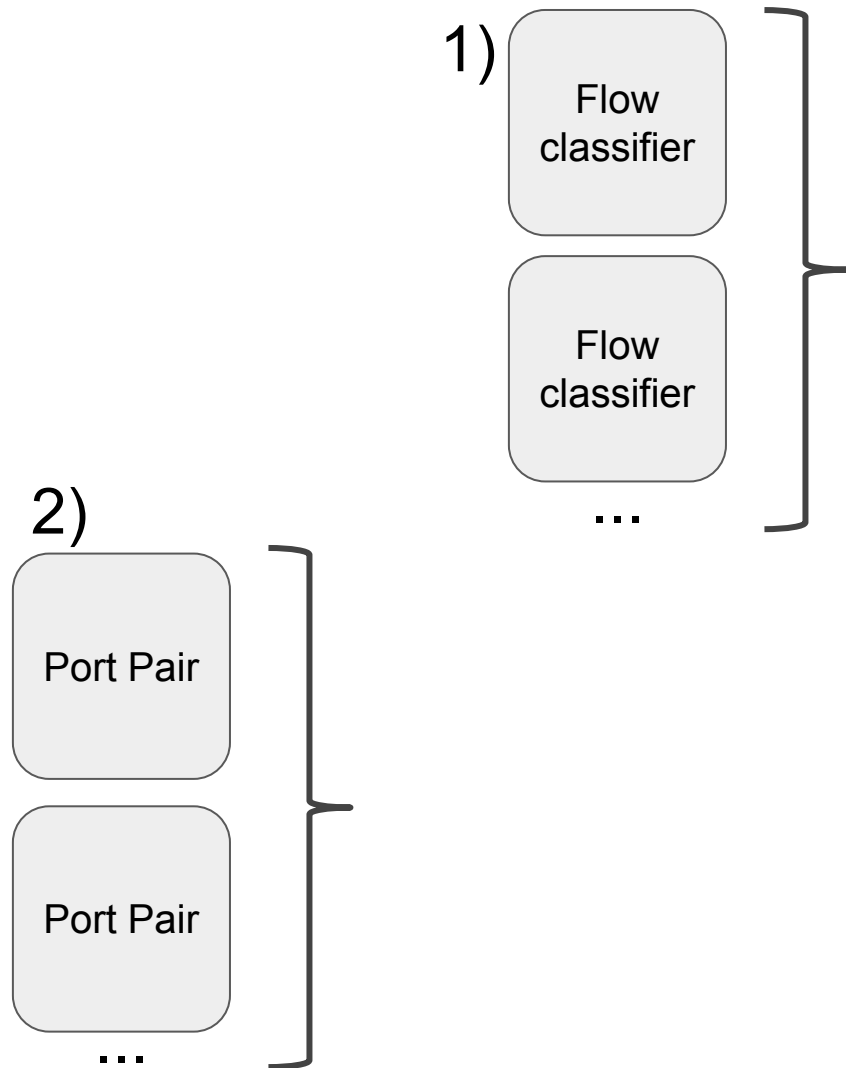
SFC-extension – Port Pair Group



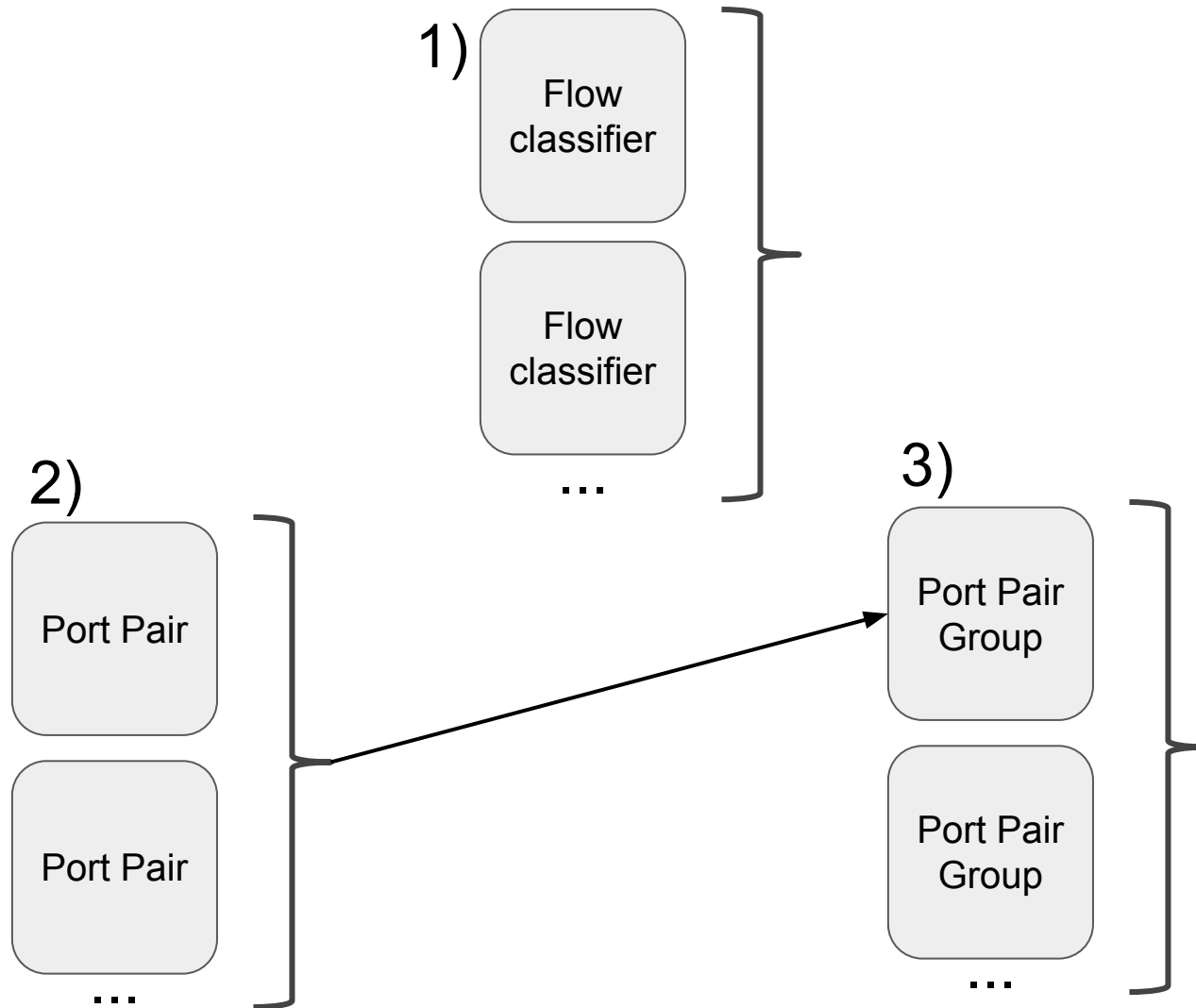
Port Chain creation with SFC-extension



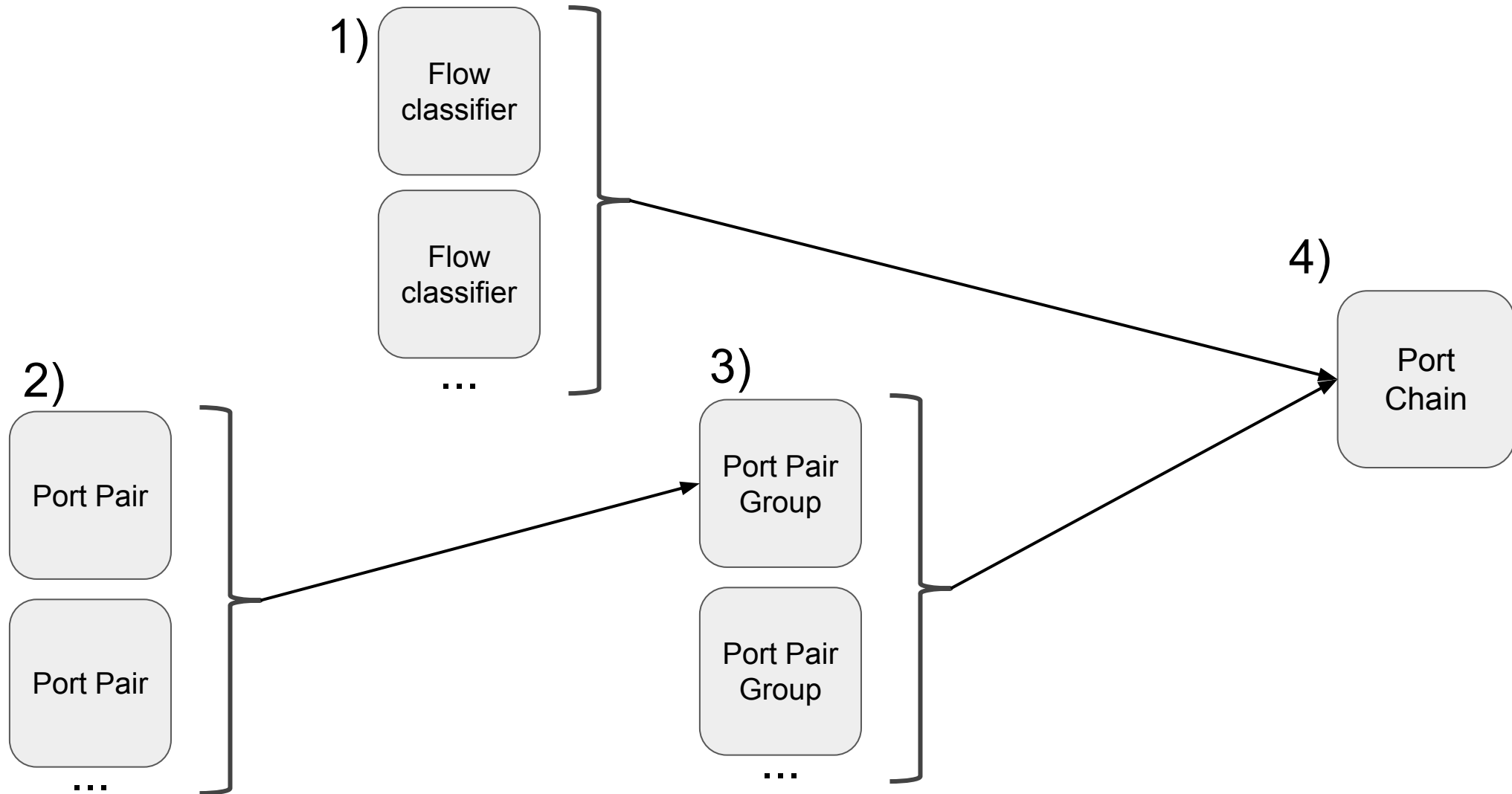
Port Chain creation with SFC-extension



Port Chain creation with SFC-extension

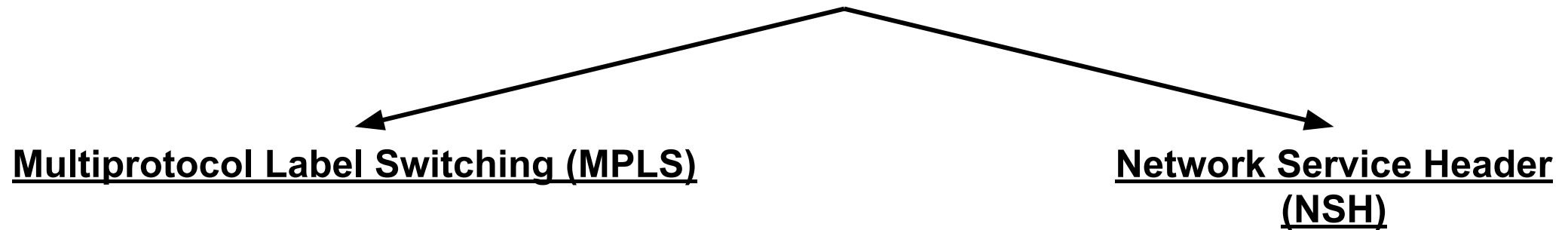


Port Chain creation with SFC-extension

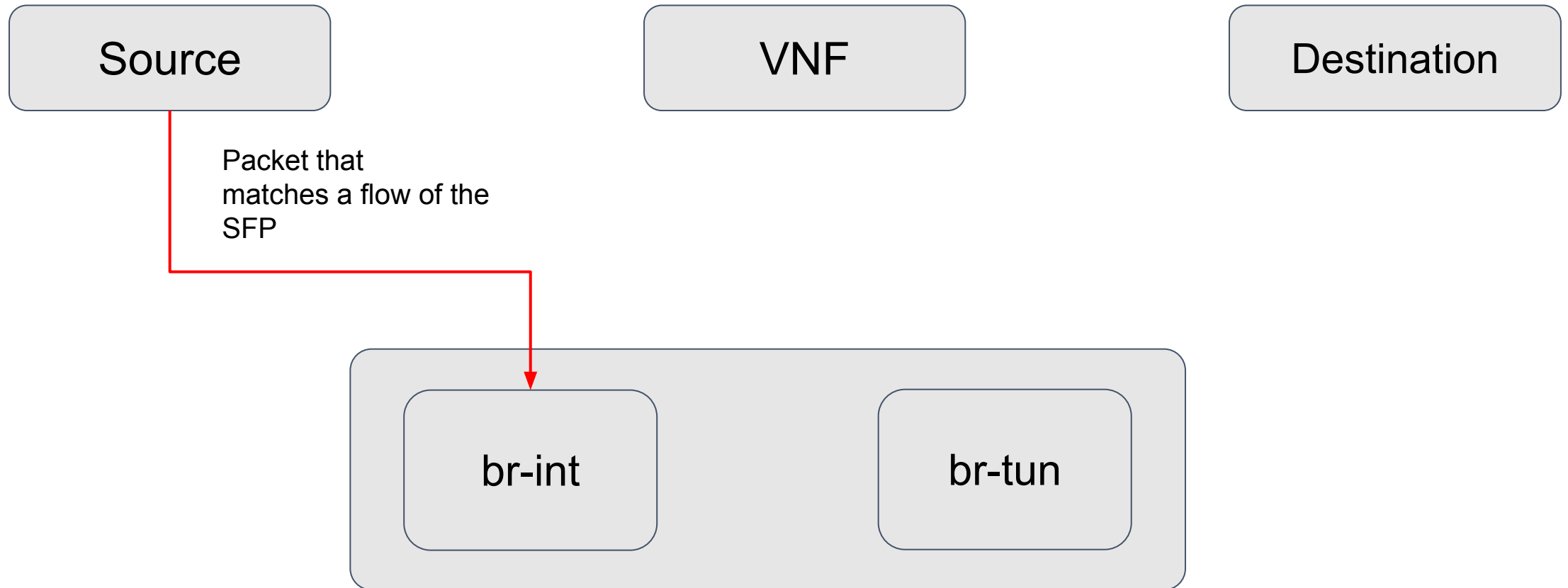


SFC-extension - Encapsulation

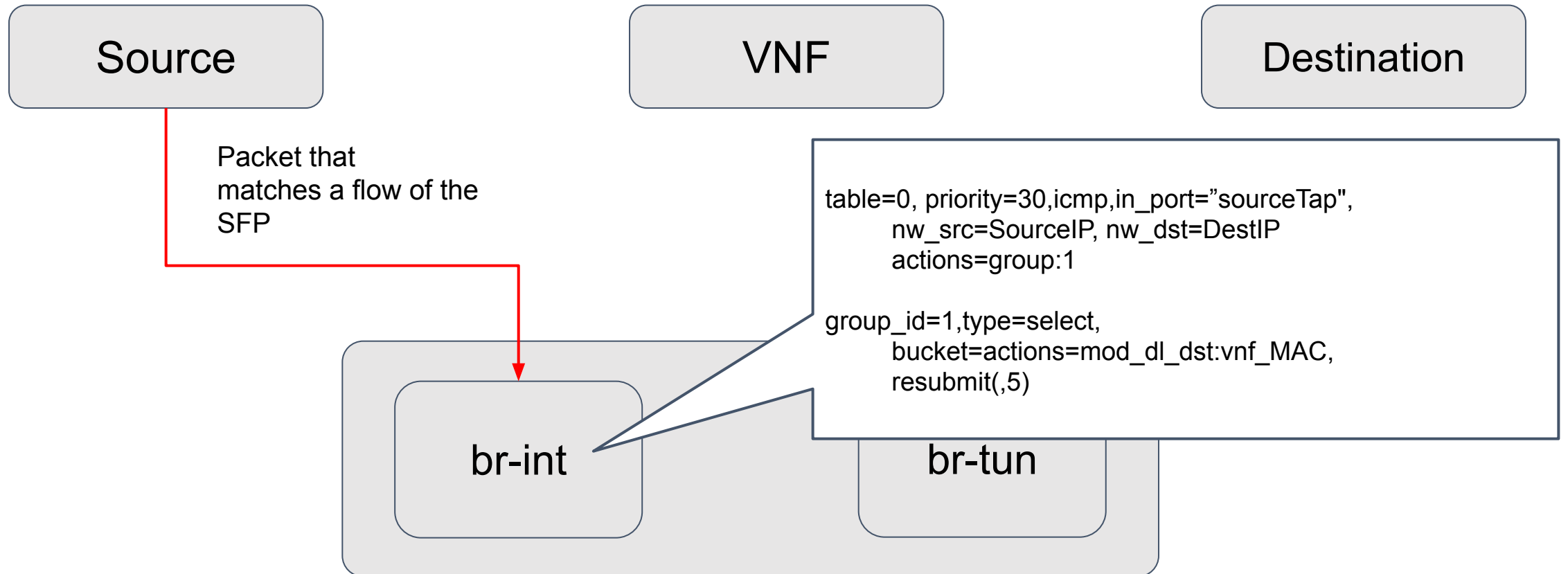
The SFC-extension support two types of encapsulation mechanism to separate traffic belonging to different Port Chain:



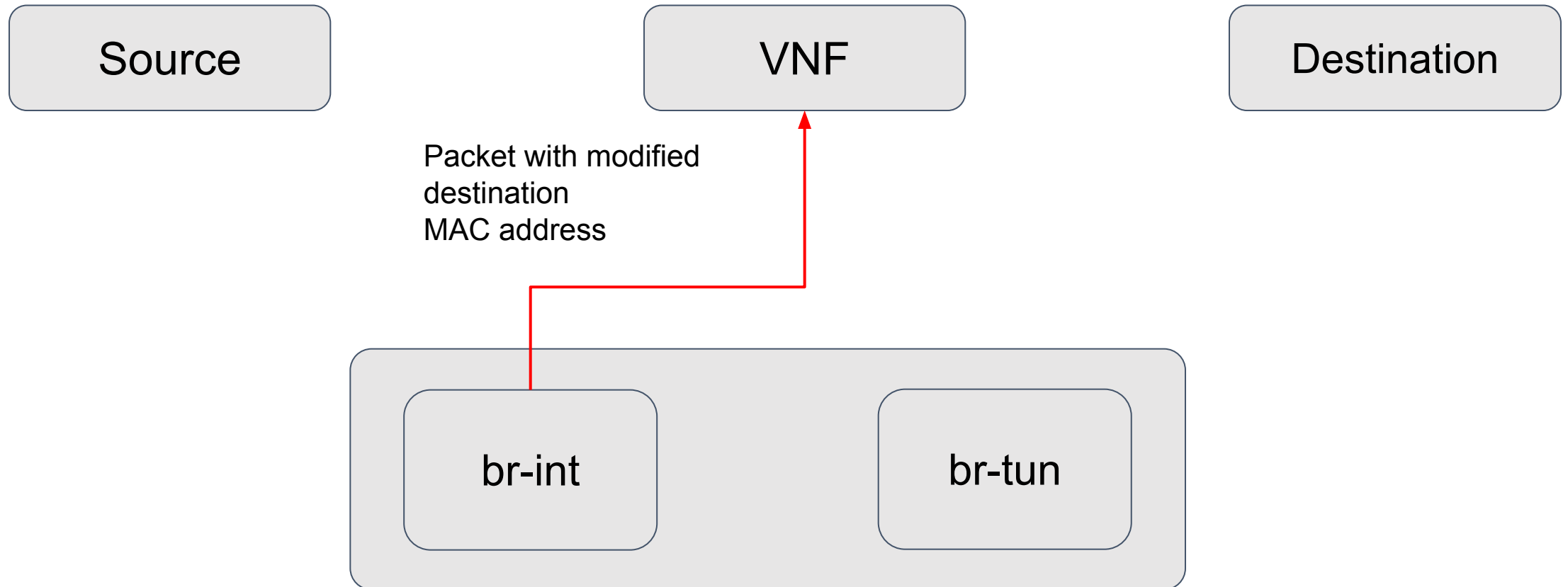
SFP deployed over a single physical node



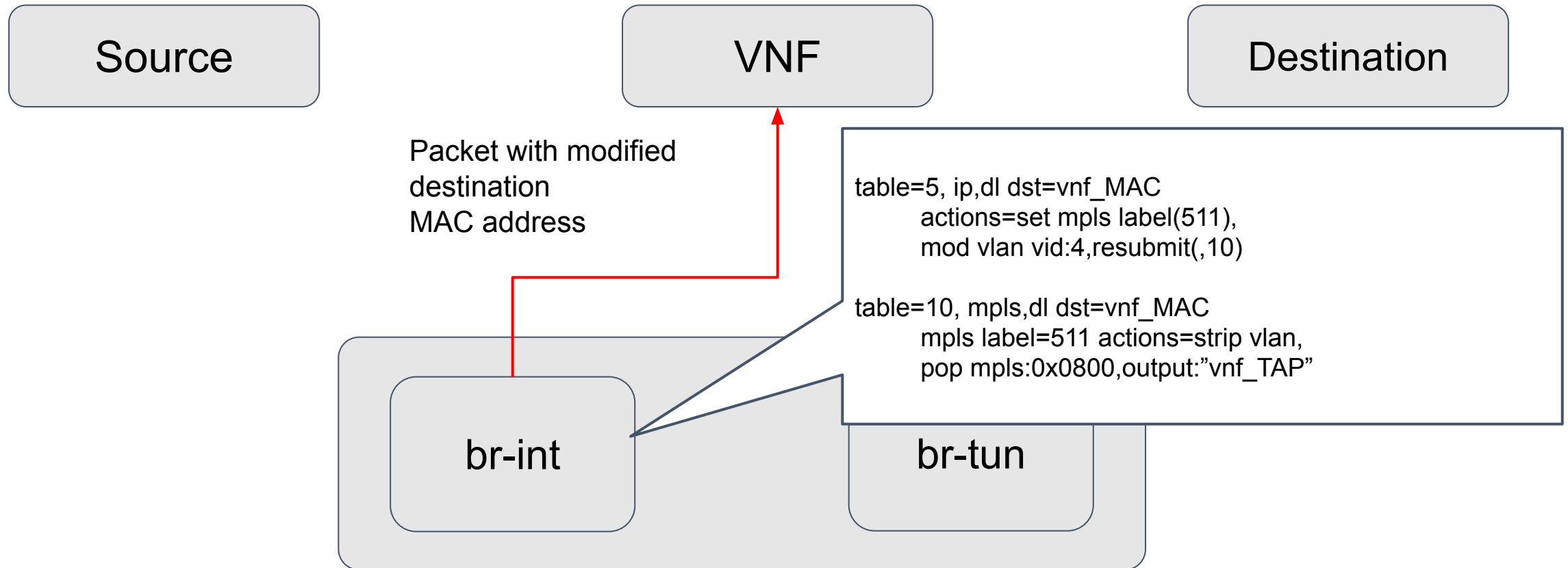
SFP deployed over a single physical node



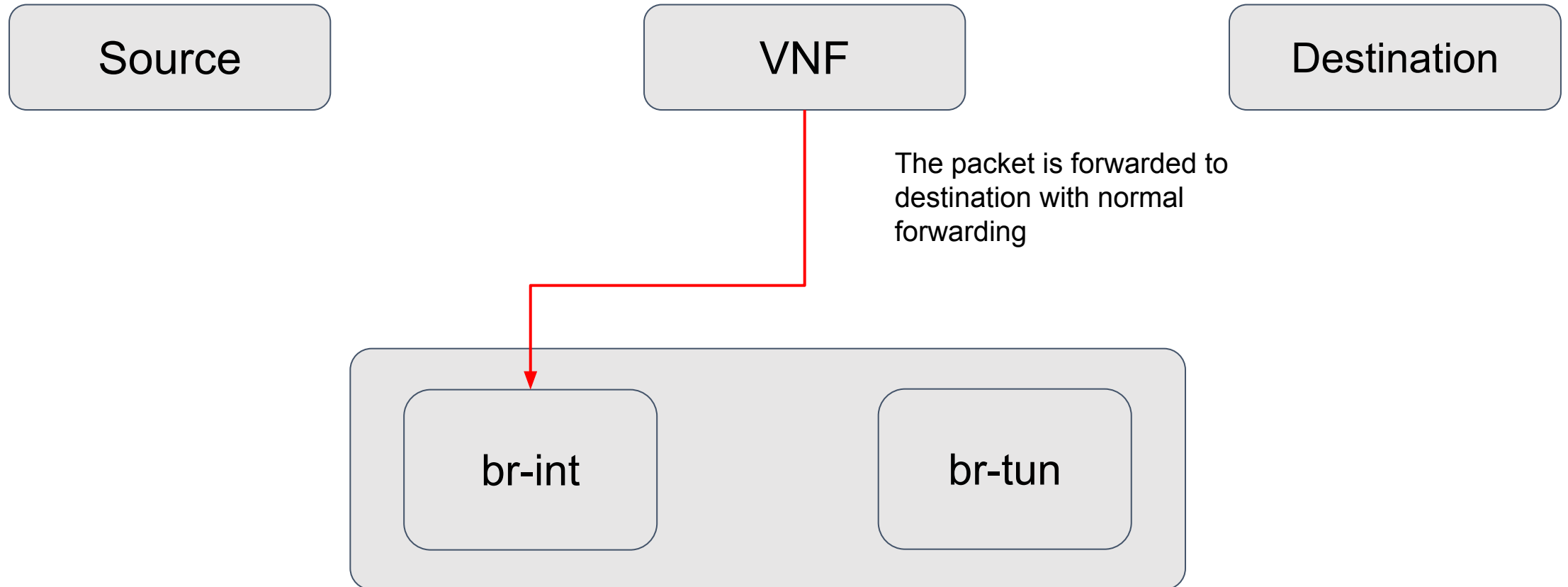
SFP deployed over a single physical node



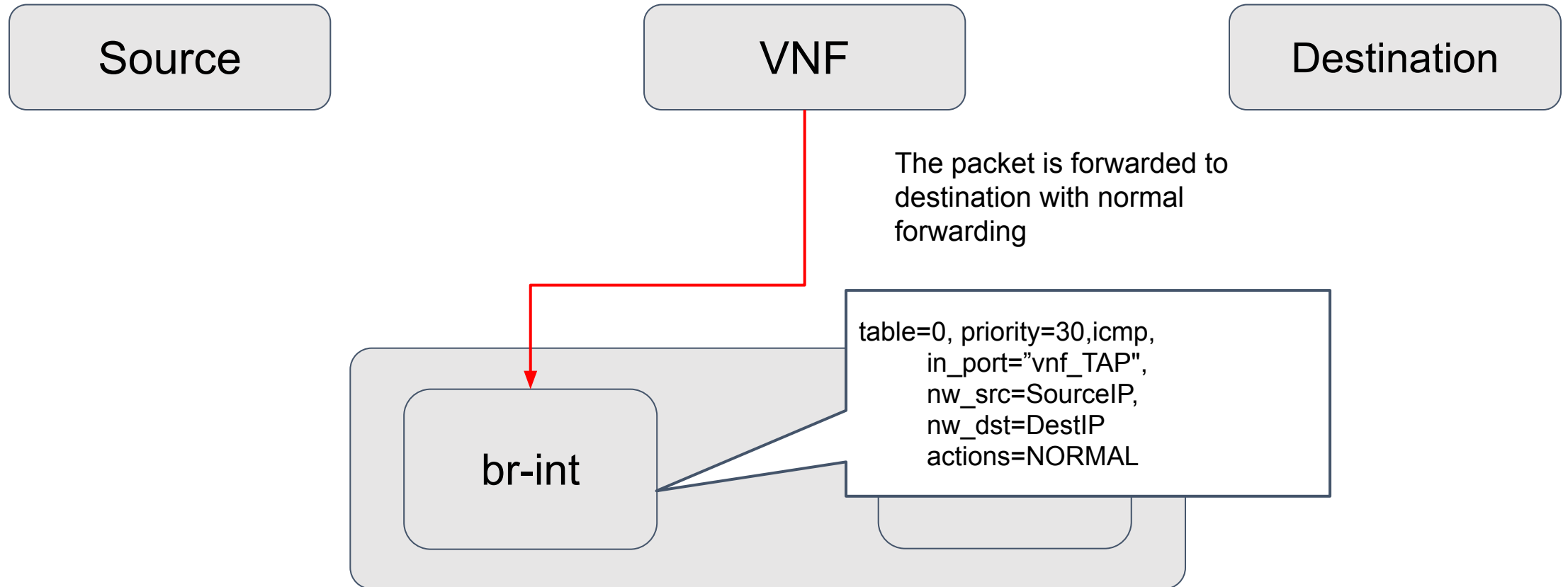
SFP deployed over a single physical node



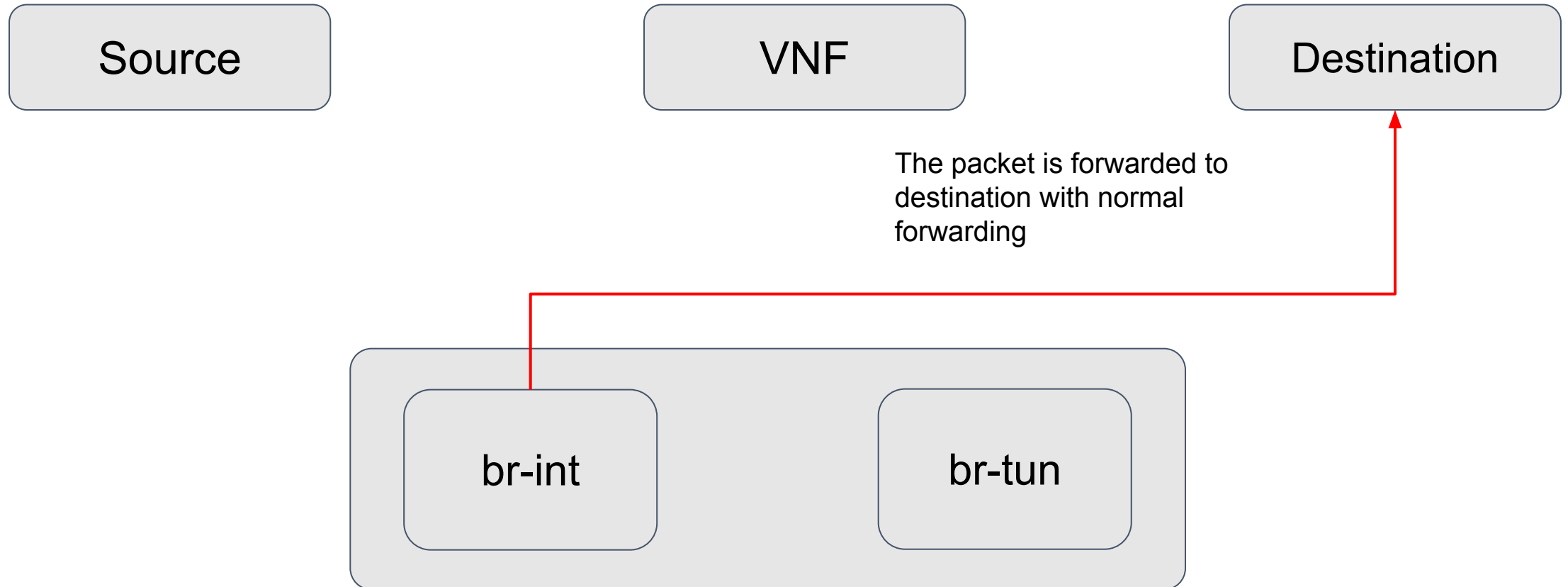
SFP deployed over a single physical node



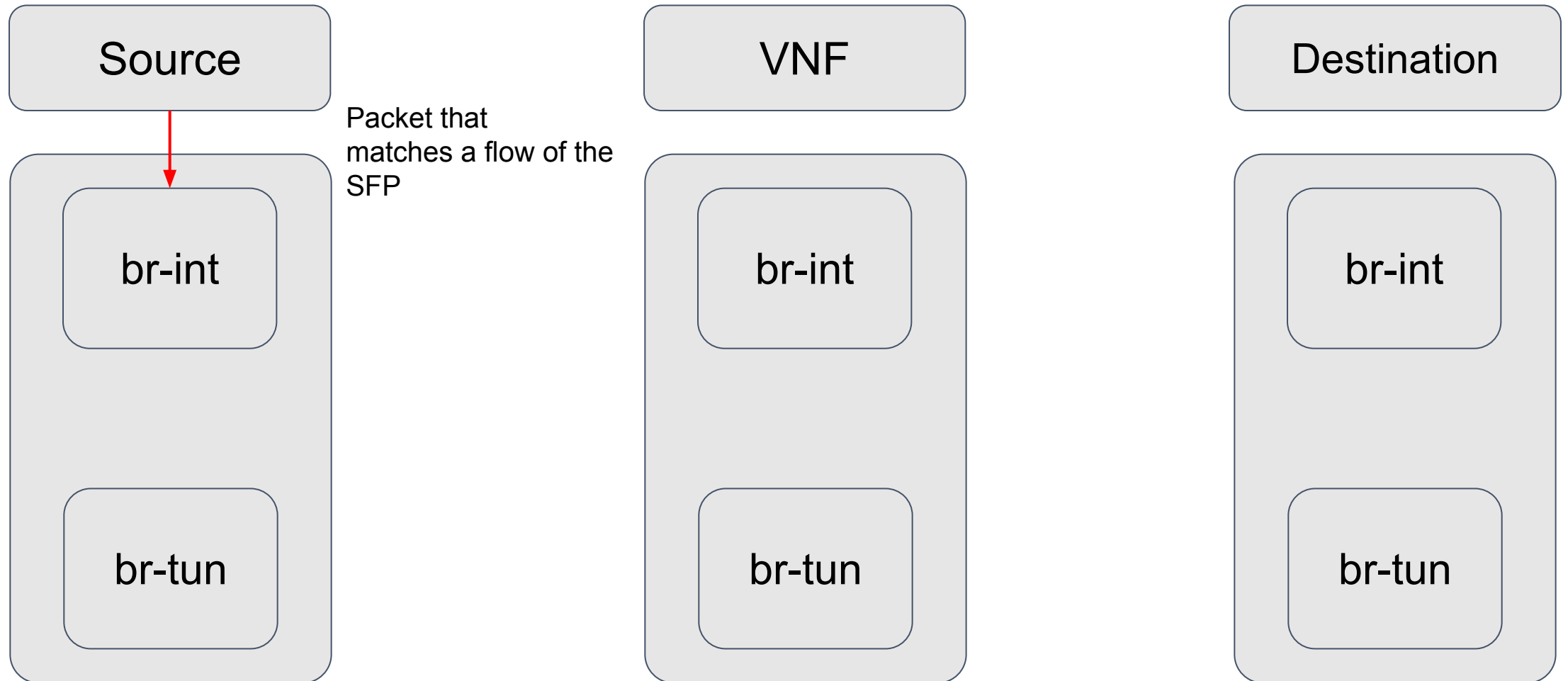
SFP deployed over a single physical node



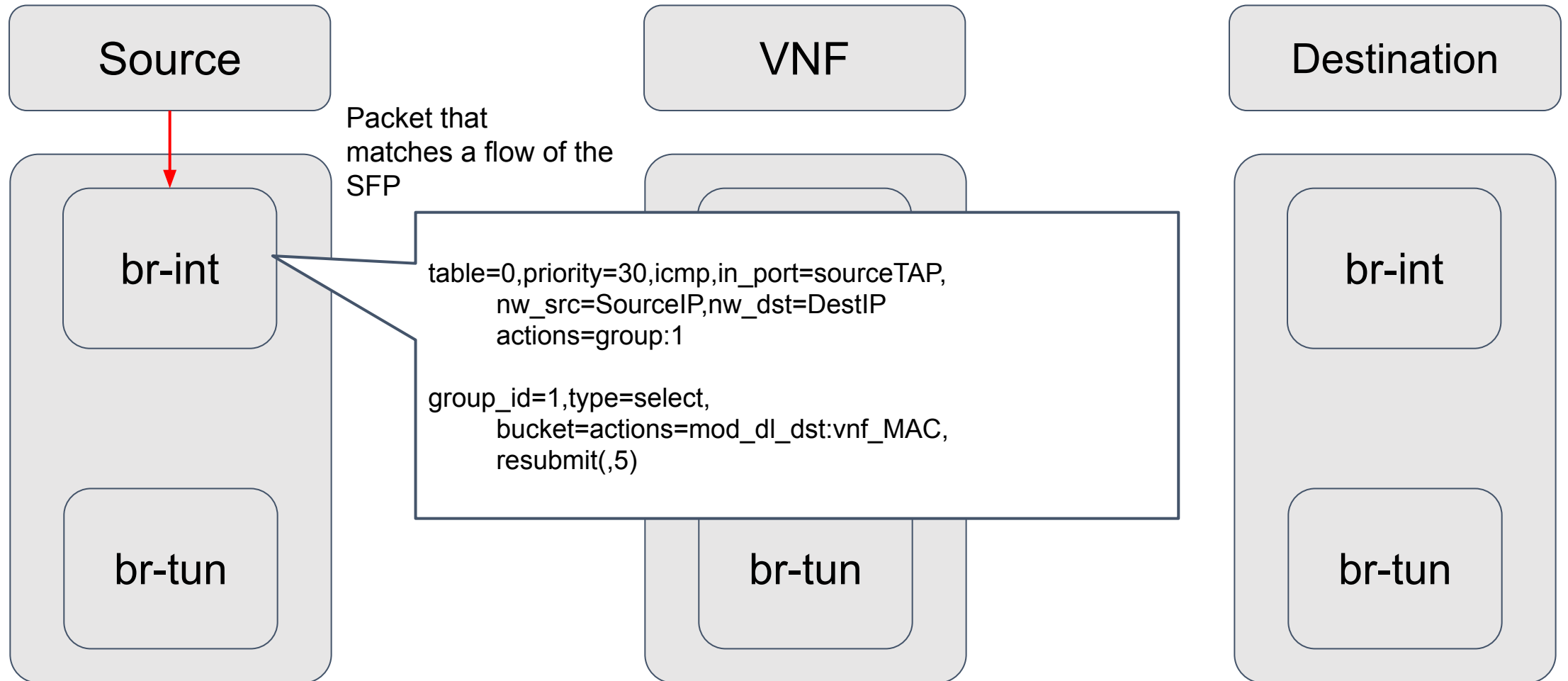
SFP deployed over a single physical node



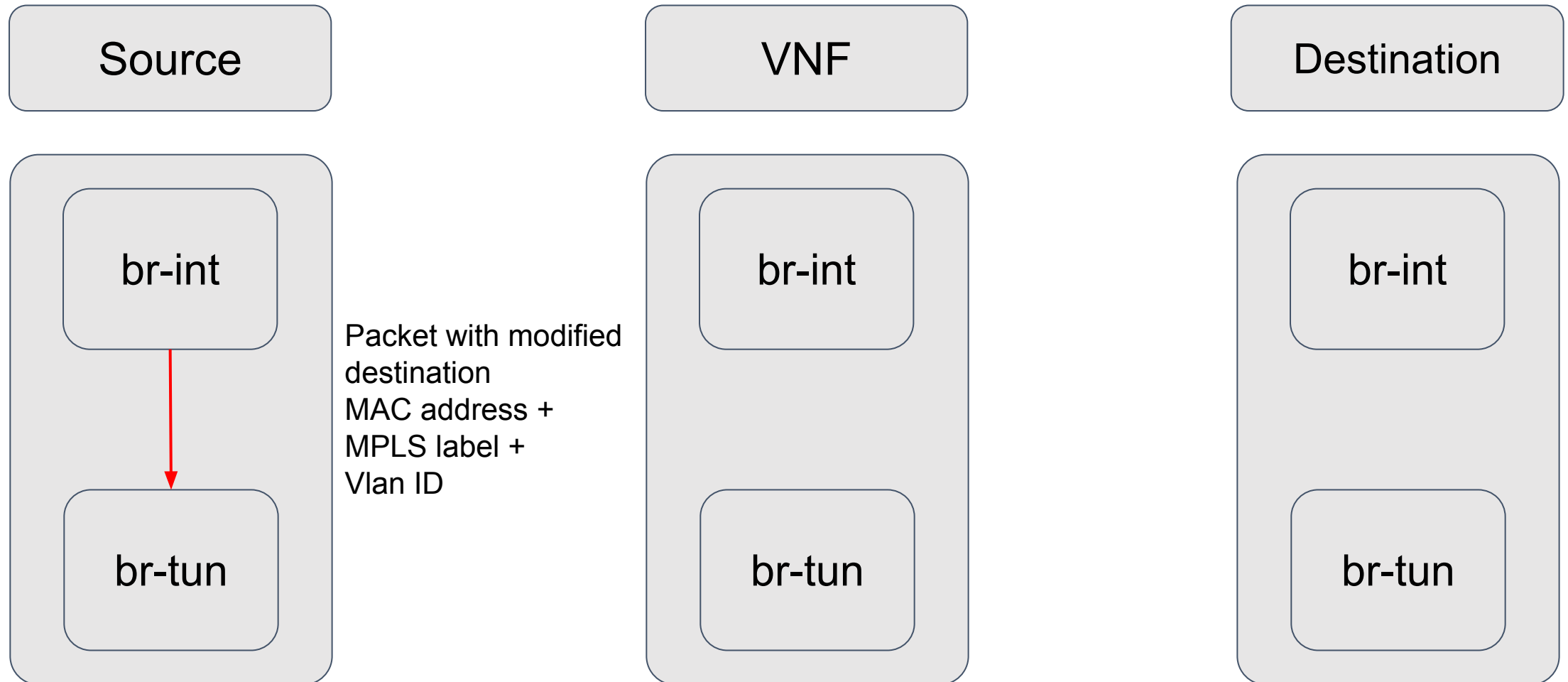
SFP deployed over multiple nodes



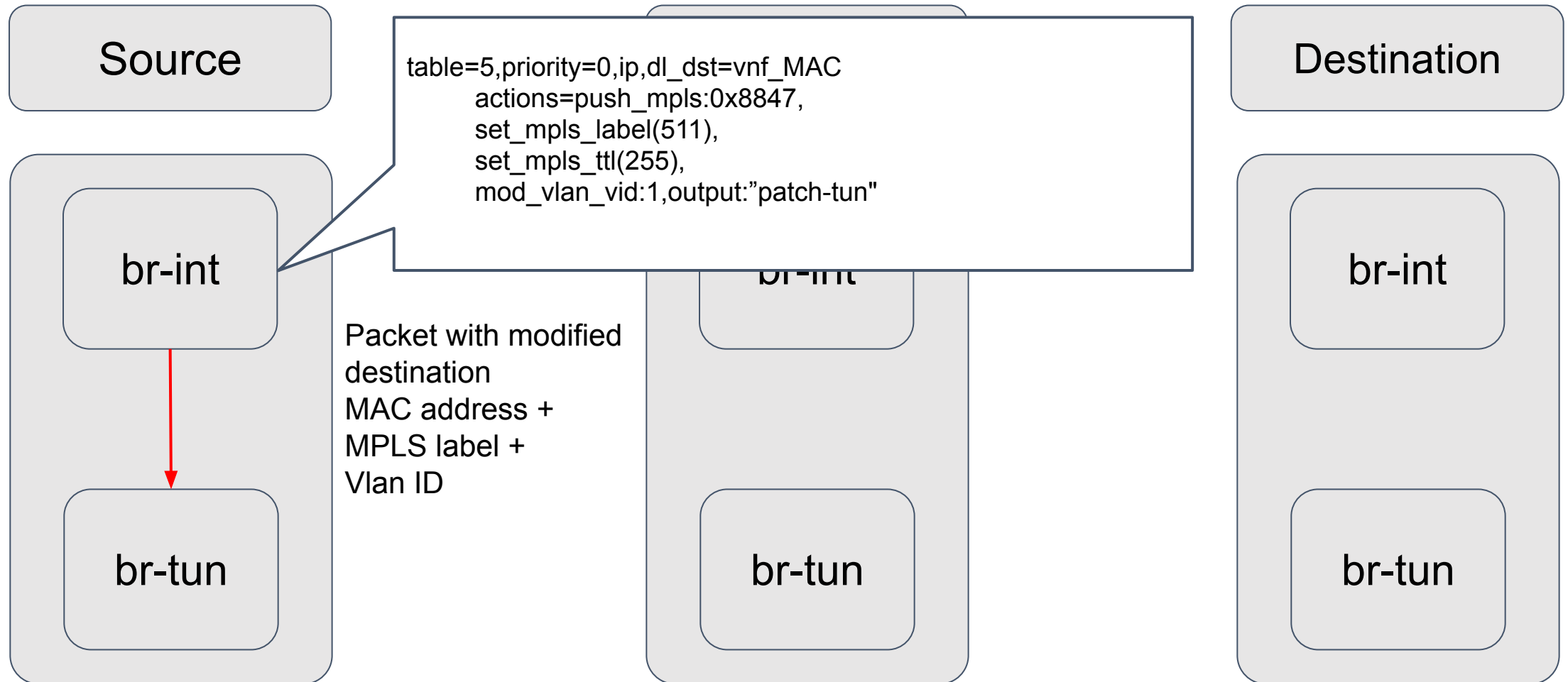
SFP deployed over multiple nodes



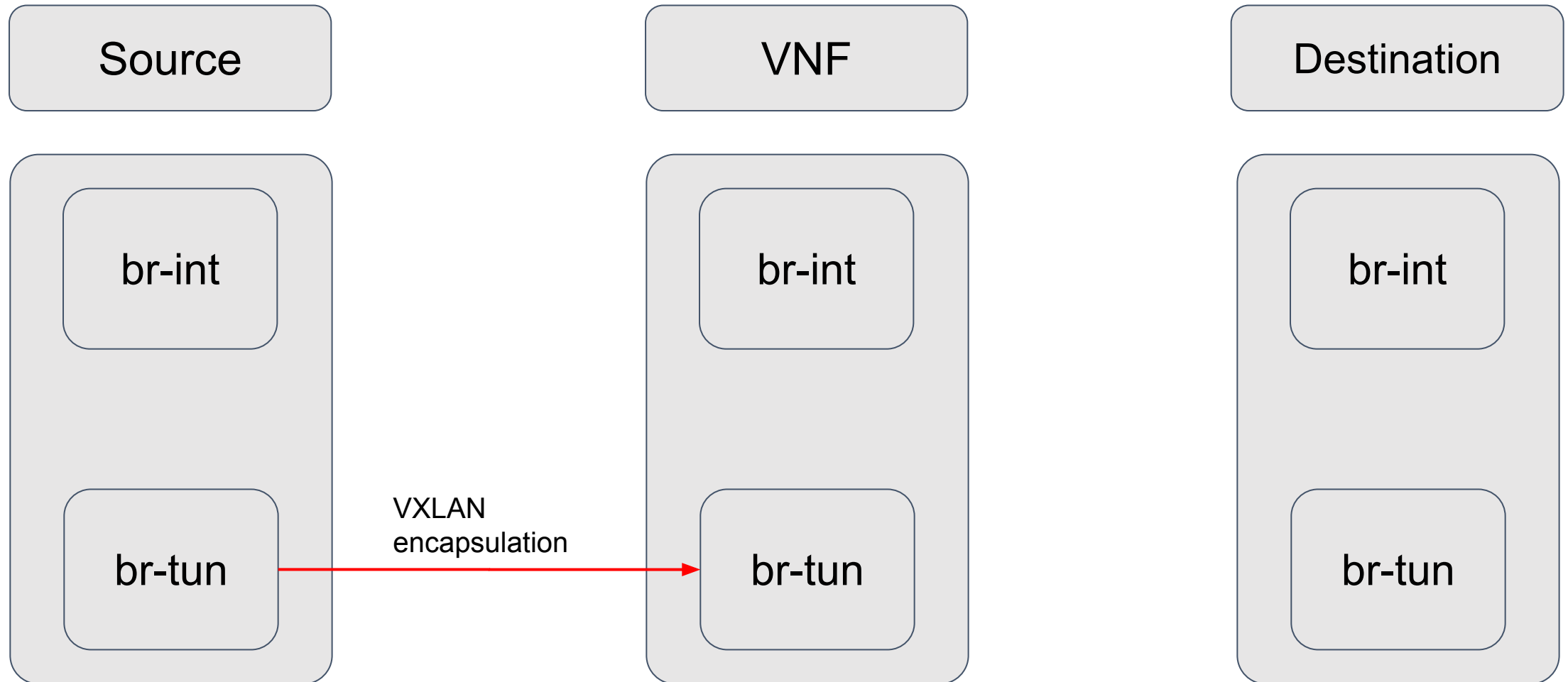
SFP deployed over multiple nodes



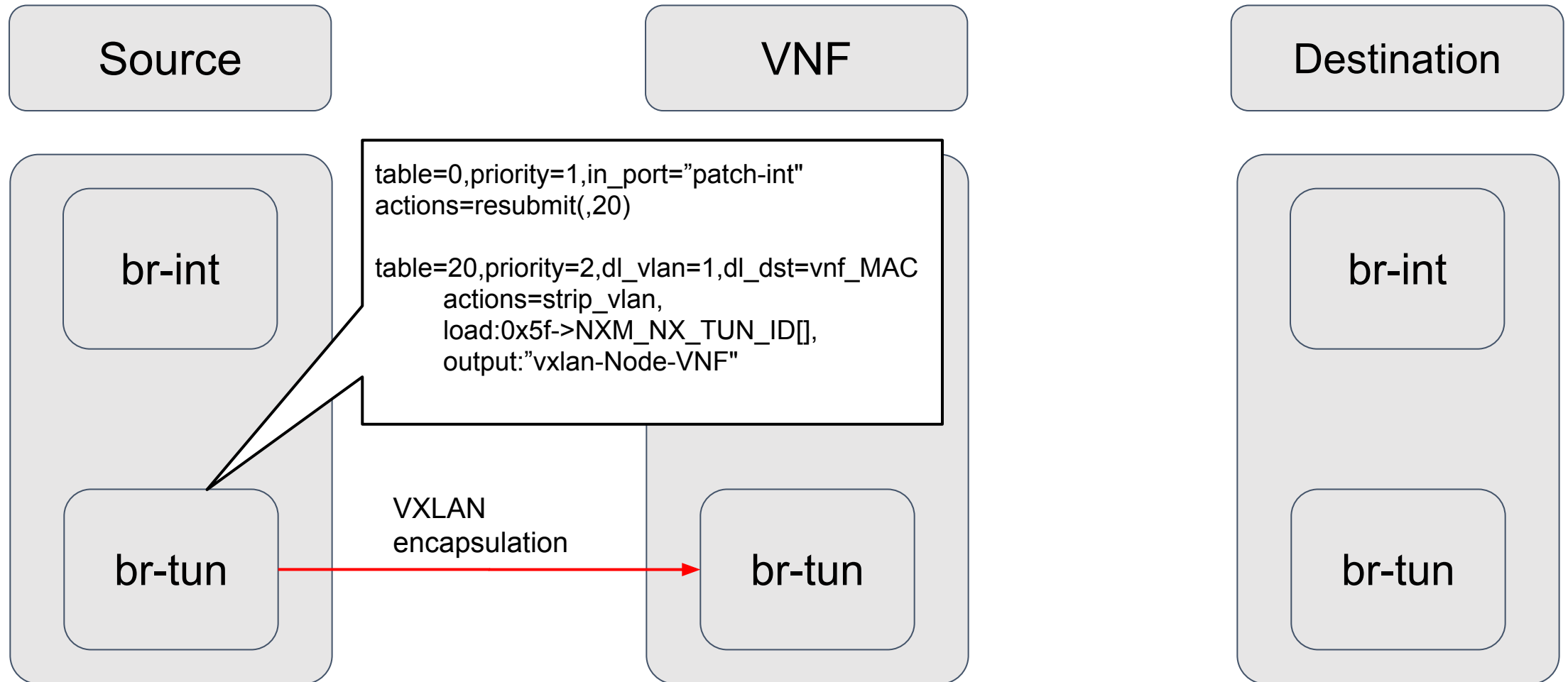
SFP deployed over multiple nodes



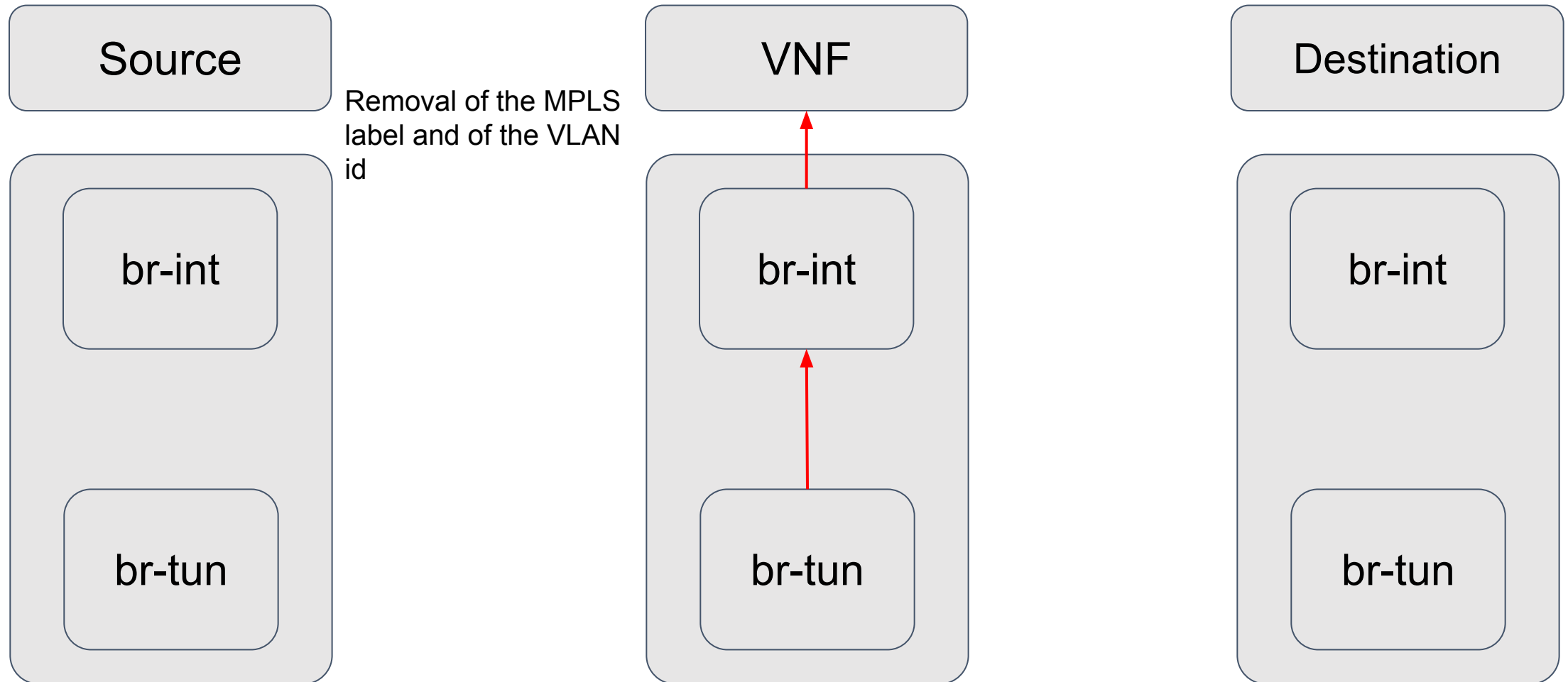
SFP deployed over multiple nodes



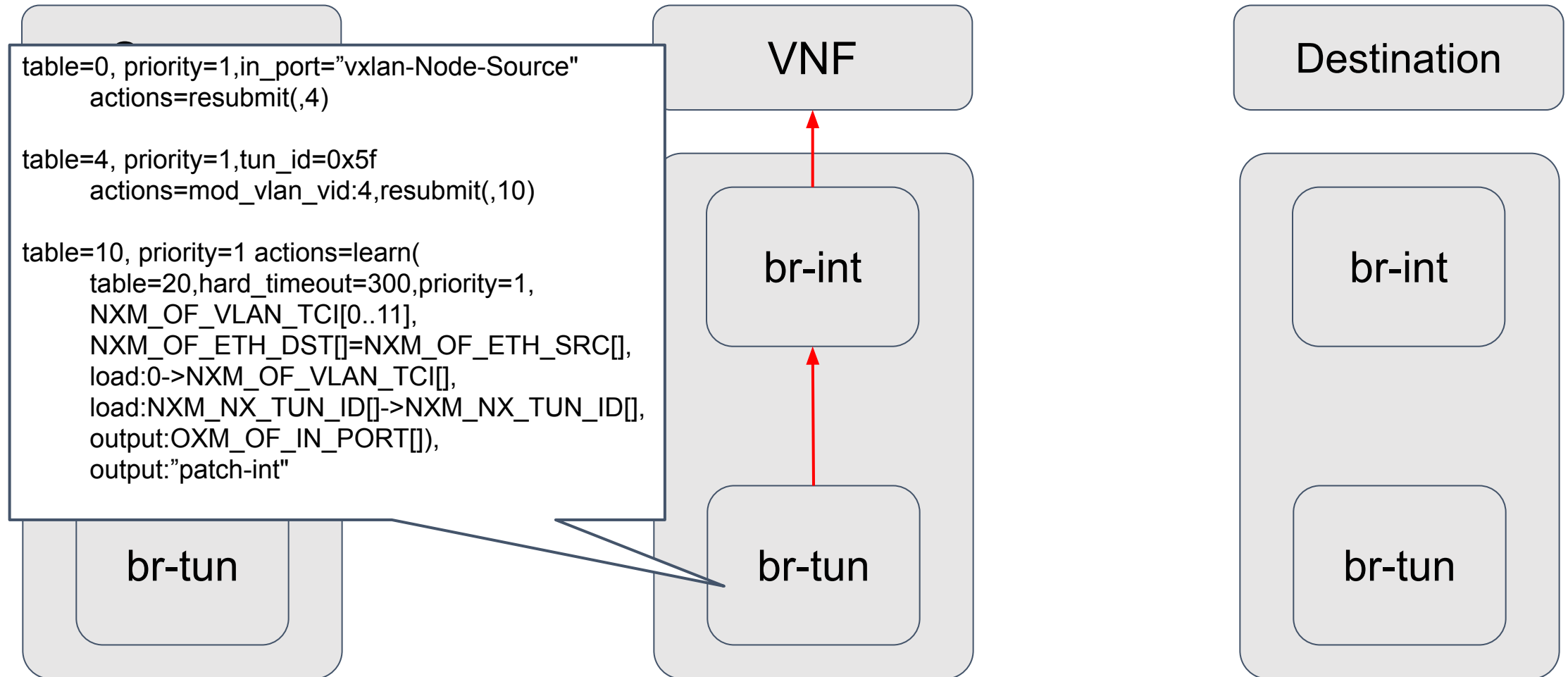
SFP deployed over multiple nodes



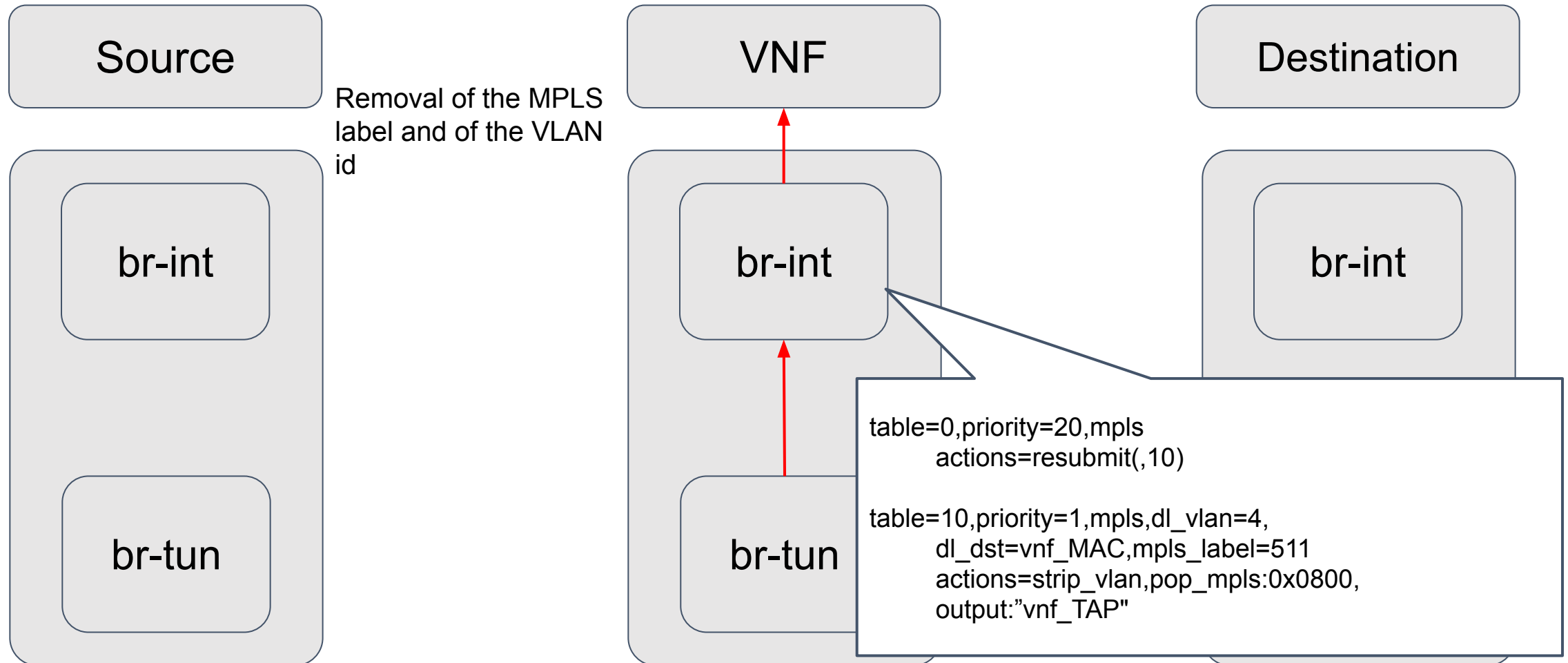
SFP deployed over multiple nodes



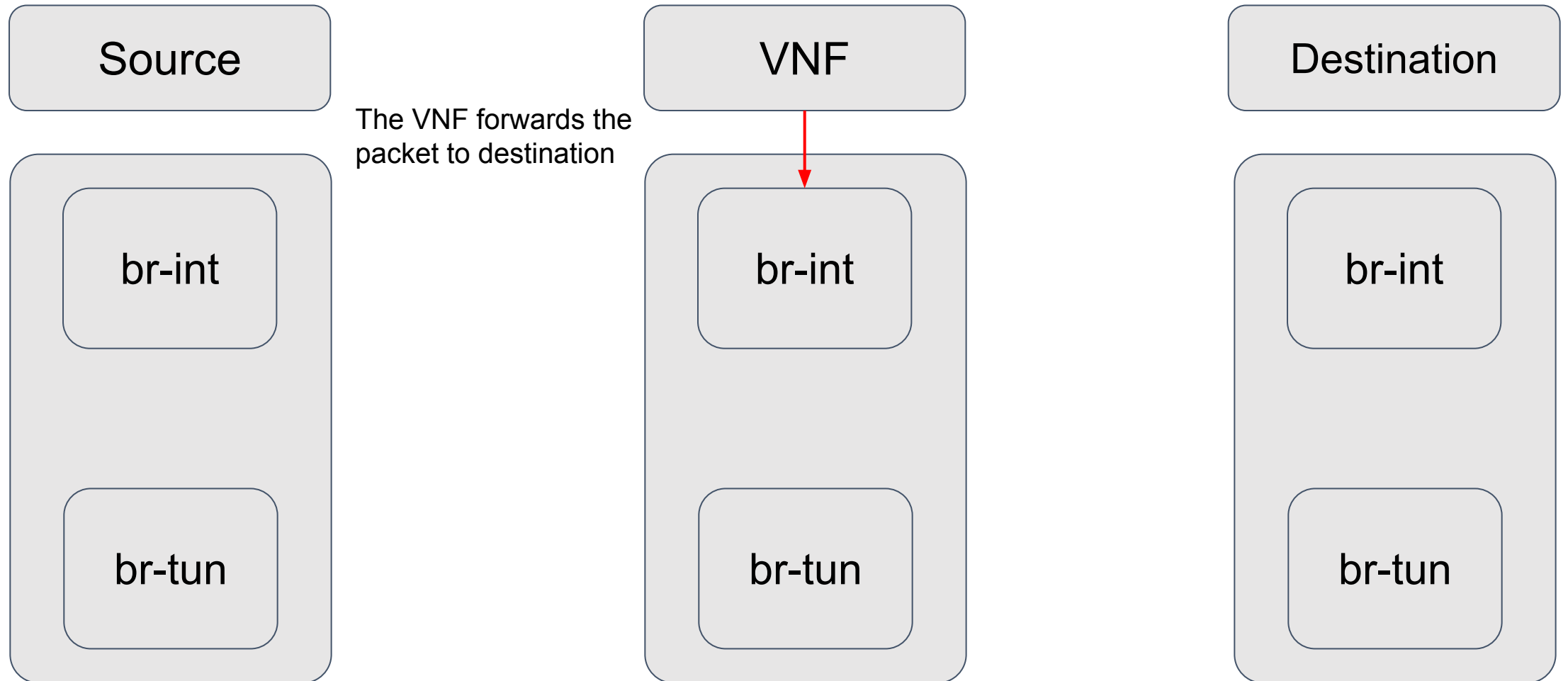
SFP deployed over multiple nodes



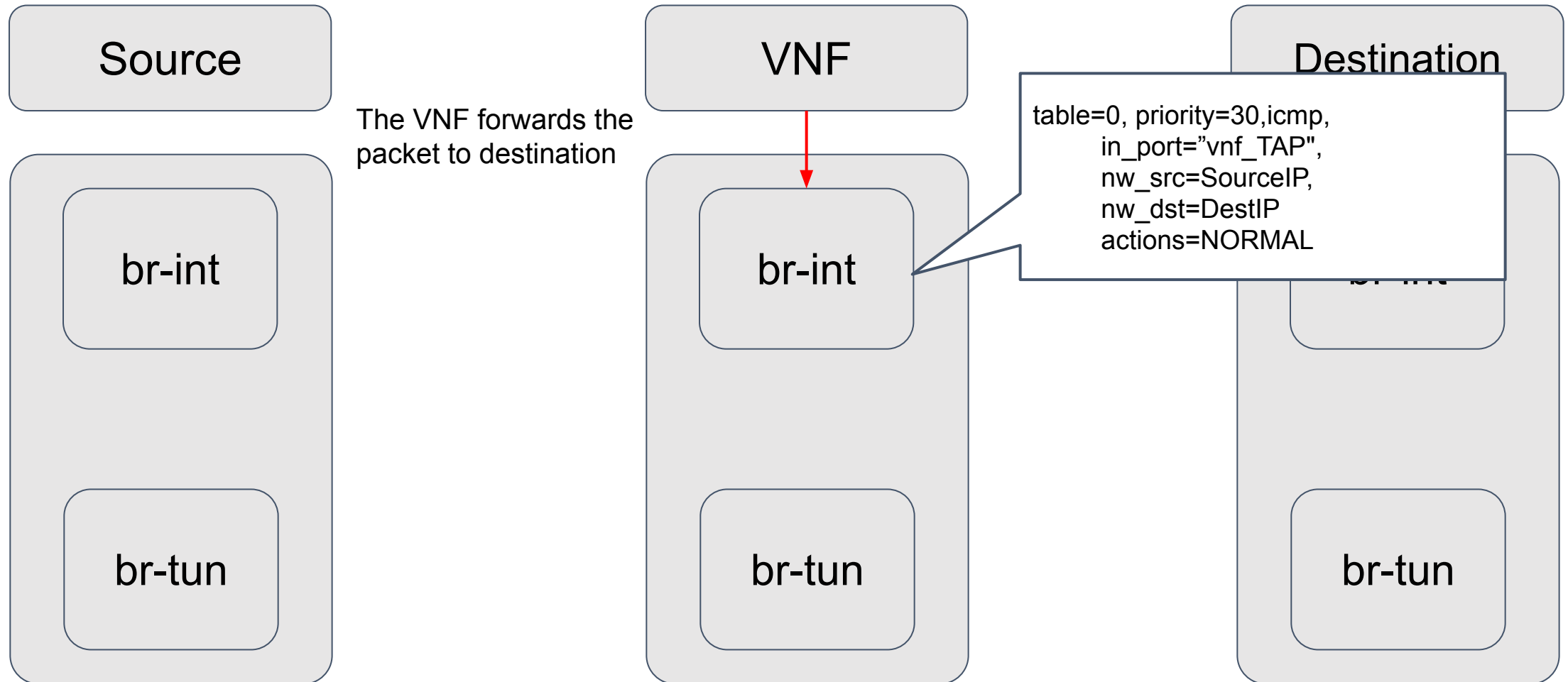
SFP deployed over multiple nodes



SFP deployed over multiple nodes



SFP deployed over multiple nodes



SFP deployed over multiple nodes

