



CUBRO
NETWORK VISIBILITY

DEDUPLICATION

WHITE PAPER

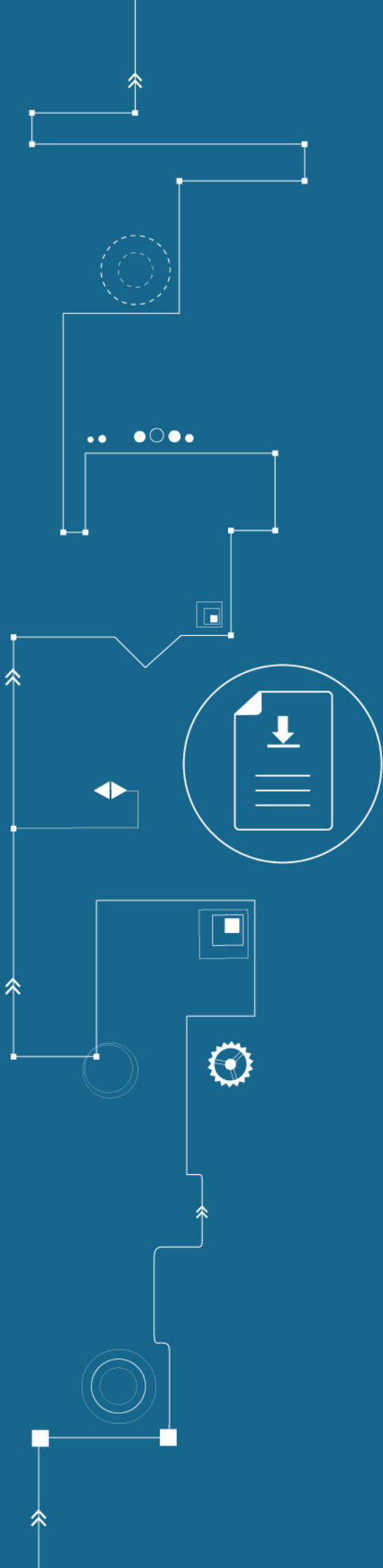
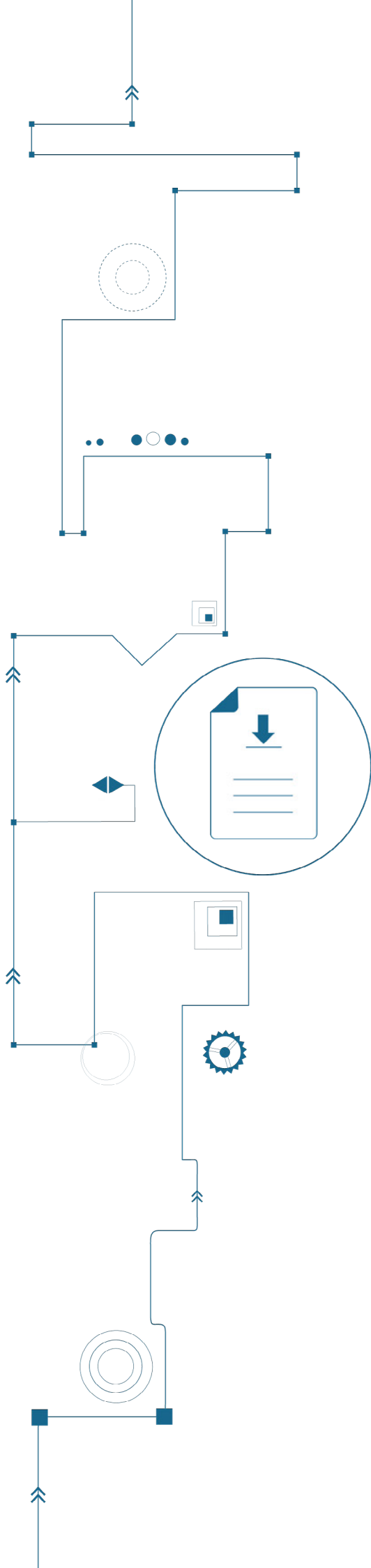




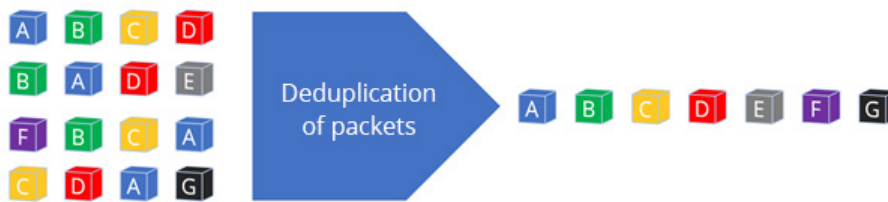
TABLE OF CONTENTS

Why Deduplication?	3
Sources of duplicated traffic	3
Traffic Flow Example	3
Different deduplication methods	4
Packet comparison based deduplication	4
Filtering based deduplication	6
Summary	9



Why Deduplication?

Network visibility attempts to capture all the packets in the network. Network visibility's task is to provide required packet streams to various monitoring tools. It may encounter duplicate packets due to overlapping tapping. Duplicate packets increase the data bandwidth and thus force the monitoring tools dimensioning to be higher. Therefore, eliminating duplicate packets will improve monitoring performance and reduce investment.



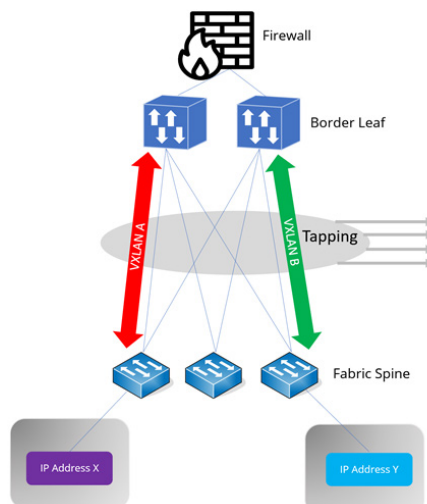
Today there is a bigger need for deduplication due to network virtualization. VXLAN is used commonly in virtualized networks. Separation of Virtual Machines (VMs) is done by using VXLAN Segment or VXLAN identifier. Only VMs within the same VXLAN segment can communicate with each other.

Each VXLAN segment is scoped through a 24-bit segment ID (VXLAN Network Identifier, VNI). This allows up to 16M VXLAN segments to coexist within the same administrative domain. Each VNI thus creates a virtual overlay network. With software-defined network it is simple to deploy new segments, because of the available large address space, and since the virtual network is decoupled from the physical. It's also easier to orchestrate the solution.

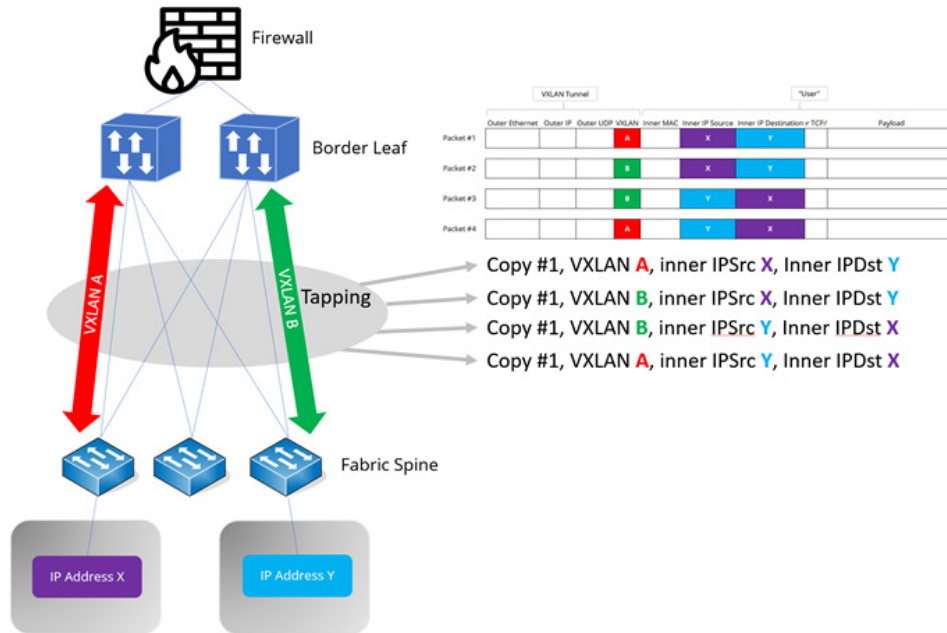
Sources of duplicated traffic

Usually duplicate traffic is caused by picking up the same traffic more than once due multiple tap and/or aggregation devices.

Traffic Flow Example



Traffic between IP X and IP Y passes the tapping two times and thus the same packets are visible twice at the tapping output. In detail, traffic on tapping output looks like this:



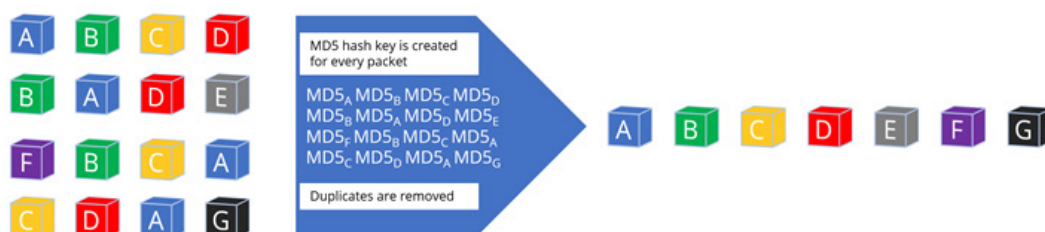
From user perspective every packet is received twice.

Different deduplication methods

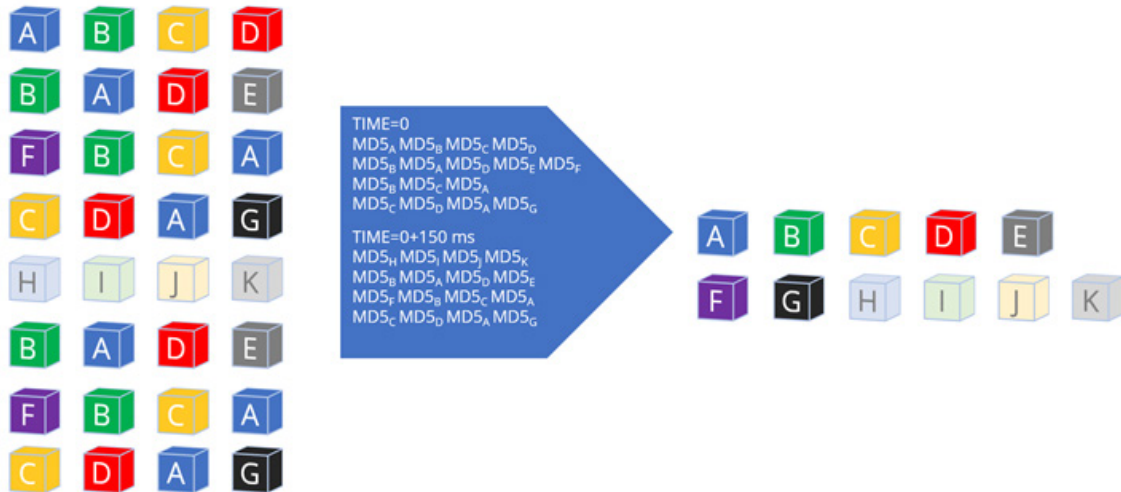
Often deduplication is understood as comparing packets to each other and removing duplicates. There are, however, two methods of removing duplicate packets and both the methods have pros and cons.

Packet comparison based deduplication

Comparing full packets against each other is not an efficient method since it would consume all the memory very fast. Instead, a MD5 checksum (hash key) is calculated for every packet. MD5 algorithm creates 128-bit hash value that identifies the packet uniquely. MD5 hash keys are compared in the memory and if a matching hash key is found it will be discarded as a duplicate. This kind of implementation often has detection window, the time in which we assume the network has provided packets from all of its domains.



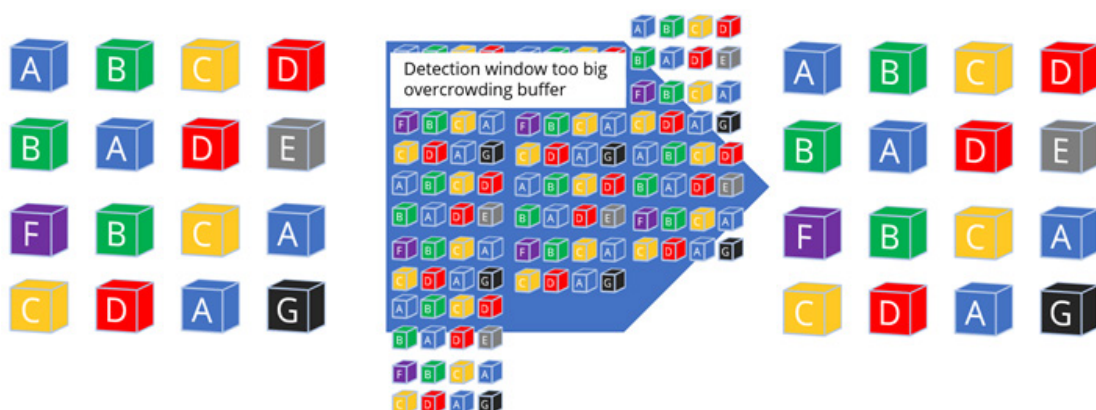
Longer detection window increases the probability of catching all the duplicate packets, but it increases the processing effort and memory usage. Some monitoring tools require control and user plane to arrive within a defined latency. If the deduplication window is too long the monitoring tool may not be able to do correlation correctly. In the example we have 16 packets at starting time and within 150 ms we get another 16 packets. If the detection window is equal or bigger than 150 ms, the dededuplication will remove duplicates within capacity limits.



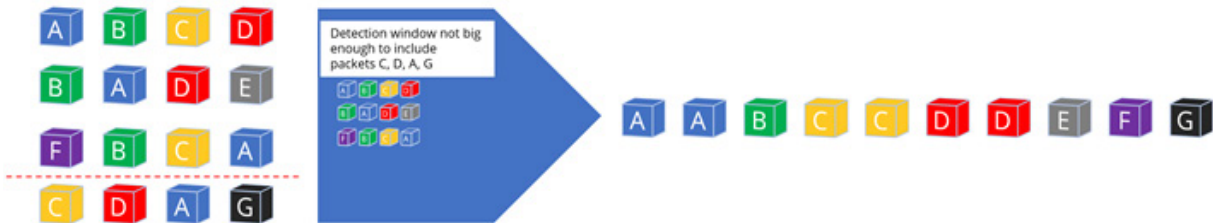
Network design, topology and implemented network elements have impact on the latency. If network latency is low, then the increase in latency caused by packet comparison may not be critical, but if the network latency is already high then additional latency can prove to be quite problematic.

Appliances doing deduplication have a buffer for the hash keys. The solution may use a fixed time window given as a parameter or automatically adjusting time window with a maximum value getting smaller when the bandwidth increases. The latter approach adjusts to the load and does better under all circumstances.

For example, if we have a buffer reserved for 300,000 hash keys, the performance of the deduplication depends on bandwidth, packet size and on the detection time. If the appliance uses a fixed time window and the bandwidth is higher than the reserved buffer the deduplication is not going to work as expected.



Automatically adjusting time window will have lower window size when the number of packets grows (bandwidth grows and/or packet size decreases) to ensure best possible deduplication performance. Of course, it will not detect duplicates if the network delivers duplicate packets outside the used time window.



With small packet size (64 B) the number of packets grows to 15M assuming 1 second detection time and 10 Gbps bandwidth. If the bandwidth is 100 Gbps for the same packet size the number of packets would be 150M. Second table shows how much buffer space in MB is needed for buffer entries / packets with the MD5 hash key within 1 second.

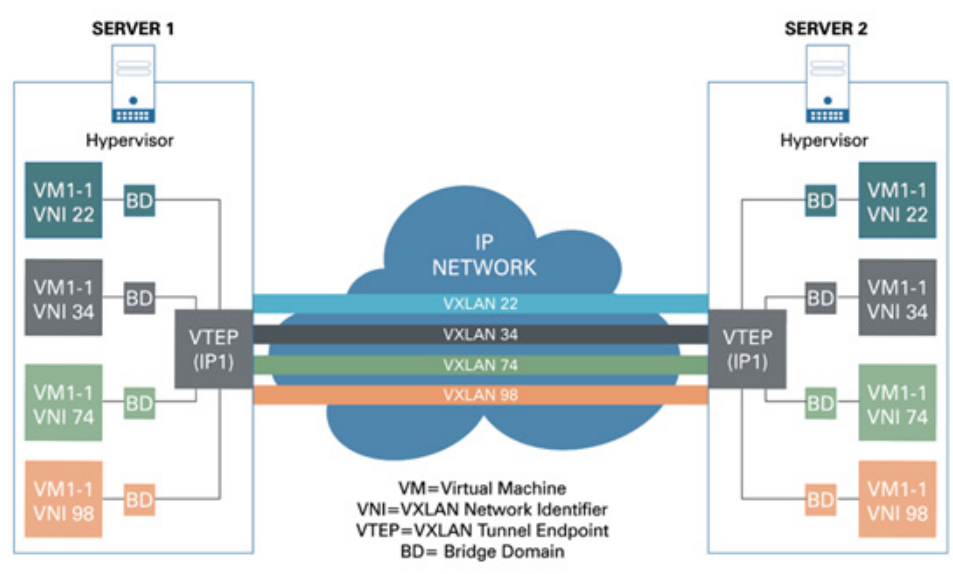
Packet /B	BW / Gbps	#Packet/s
64	10	14,880,952
300	10	3,906,250
500	10	2,403,846
1000	10	1,225,490
1518	10	763,125

Hash key /B	Buffer entries	Mem / MB
128	300,000	37
128	2,500,000	305
128	15,000,000	1,831
128	150,000,000	18,311

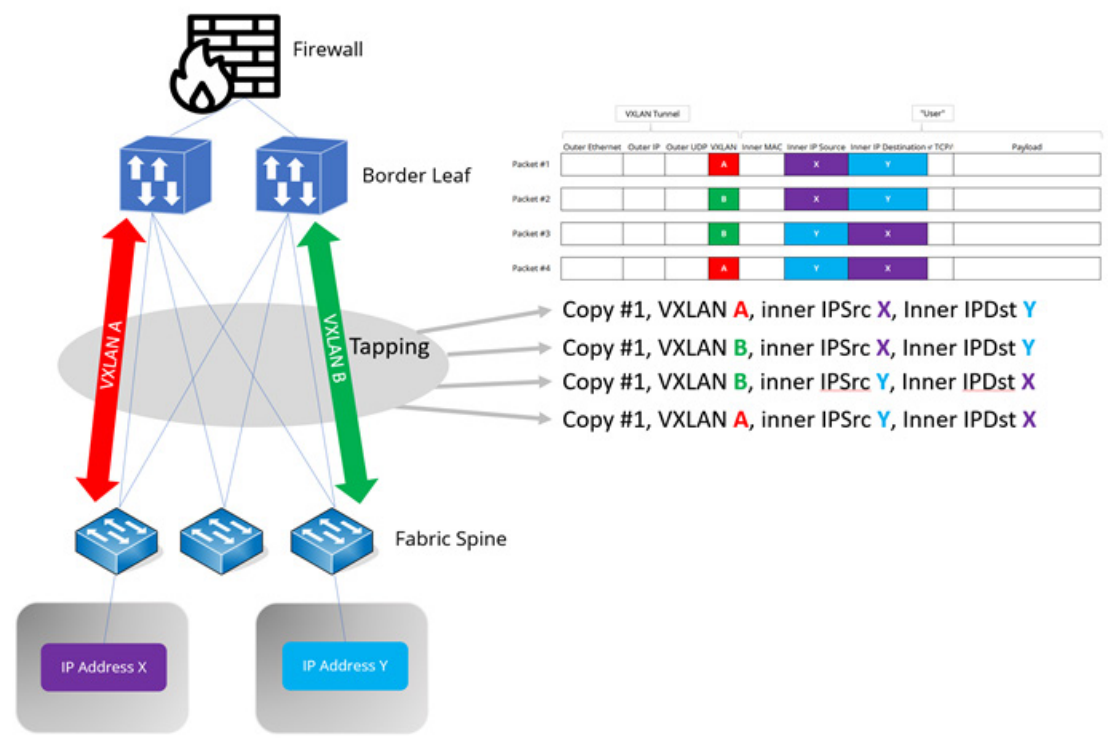
This example calculation (packet overhead included) shows clearly that although packet comparison is an effective method for dropping duplicate packets, it has drawbacks related to the required appliance resources and latency.

Filtering based deduplication

Modern sizeable networks are using overlay techniques to manage traffic flows. VXLAN is a network virtualization technology that attempts to address the scalability problems associated with large cloud computing deployments. In data centers, VXLAN is the most commonly used protocol to create overlay networks that sit on top of the physical network, enabling the use of a virtual network of switches, routers, firewalls, load balancers, and so on.



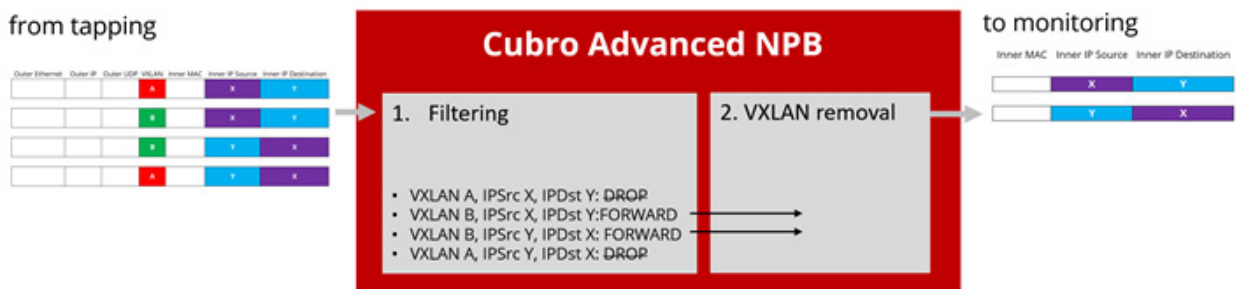
Depending on the traffic flow and tapping concept a portion of the traffic in data centers will be redundant. As described in above example a possible call flow scenario is:



An easy and straight-forward way to eliminate the duplicates is to use a combination of **VXLAN VNI** and **inner IP filtering**.

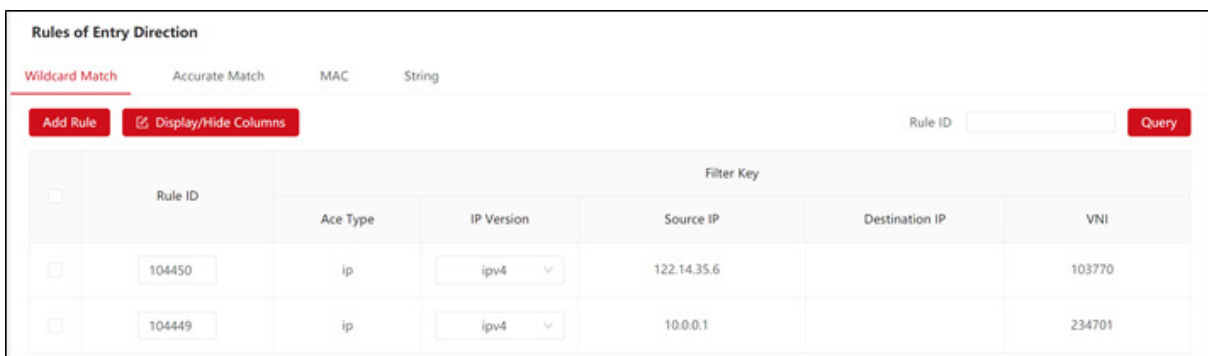
	Outer Ethernet	Outer IP	Outer UDP	VXLAN	Inner MAC	Inner IP Source	Inner IP Destination	Network Broker Probe ACTION
Packet #1				A		X	Y	drop
Packet #2				B		X	Y	forward
Packet #3				B		Y	X	forward
Packet #4				A		Y	X	drop

The advanced Packet Broker enables all the required filtering (VXLAN & inner IP) and also removes the outer VXLAN tunnel before sending the packets to the monitoring system. This way output traffic to the monitoring system does not include any duplicates and has pure “user-data” without VXLAN encapsulation.

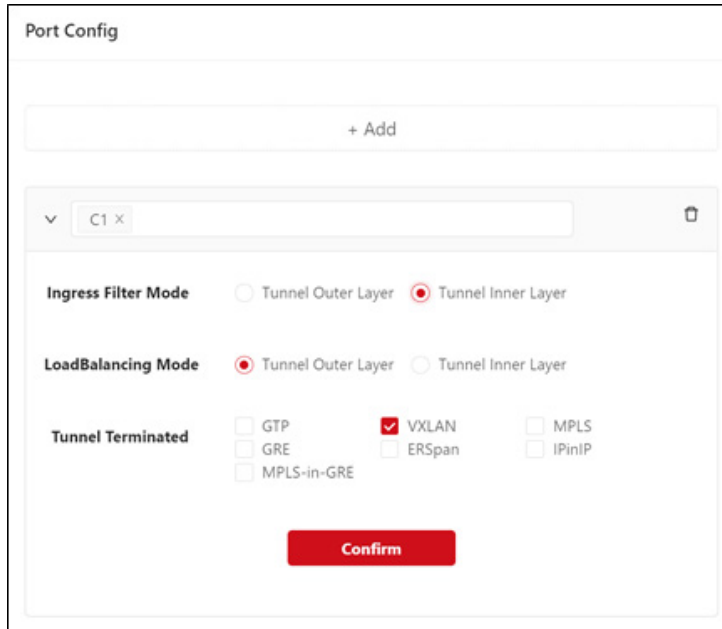


From user perspective the setup of the filtering and VXLAN removal should be as simple as possible. Following screenshots from Cubro Advanced NPBs show the easy and fast setup for filtering and VXLAN removal.

VXLAN VNI and inner IP filtering:



VXLAN removal – by one click all VXLAN tunnels are removed:



The screenshot shows a 'Port Config' window with a '+ Add' button at the top. Below it is a dropdown menu showing 'C1 X'. The configuration options are as follows:

- Ingress Filter Mode:** Tunnel Outer Layer, Tunnel Inner Layer
- LoadBalancing Mode:** Tunnel Outer Layer, Tunnel Inner Layer
- Tunnel Terminated:**
 - GTP
 - VXLAN
 - MPLS
 - GRE
 - ERSpan
 - IPinIP
 - MPLS-in-GRE

A red 'Confirm' button is located at the bottom center of the configuration panel.

Filter based deduplication is very efficient and it doesn't require additional resources if the NPB supports it. The downside is that the network topology needs to be known.

Summary

Both deduplication methods, packet comparison based and filtering have their pros and cons.

In small networks the packet comparison may be the easiest solution, but in big networks its use needs to be carefully evaluated in order to control costs and latency.

Filter based deduplication requires knowledge about the network topology, but it provides an efficient and well scaling solution for networks.