October 05, 2017

Hashing technique to optimally balance load within switching networks

Junlan Zhou

Zhengrong Ji

Hashing technique to optimally balance load within switching networks

ABSTRACT

Network protocols use multiple paths to route packets between network nodes, such that network capacity is optimally utilized in a scalable manner. Equal cost multipath (ECMP) and weighted cost multipath (WCMP) are routing techniques to balance load across multiple paths between traffic source and destination. Load balancing is typically achieved using a hash function to select the next hop for flows of balanced size. Hash functions are provided by switch hardware, and there is often only a limited number of uncorrelated hash functions. Reuse of correlated hash functions can lead to imbalanced traffic distribution within the network. The present disclosure describes techniques to balance traffic distribution within a network by use of VLAN (virtual local area network) tags as a hash input and by shuffling VLAN tags of packets at switches along a path.

KEYWORDS

- ECMP
- WCMP
- Hash function
- Load balancing
- Traffic polarization

BACKGROUND

Reuse of hash functions amongst switches within a network can result in an imbalanced traffic load ("traffic polarization"). This is illustrated by the example of Fig. 1.

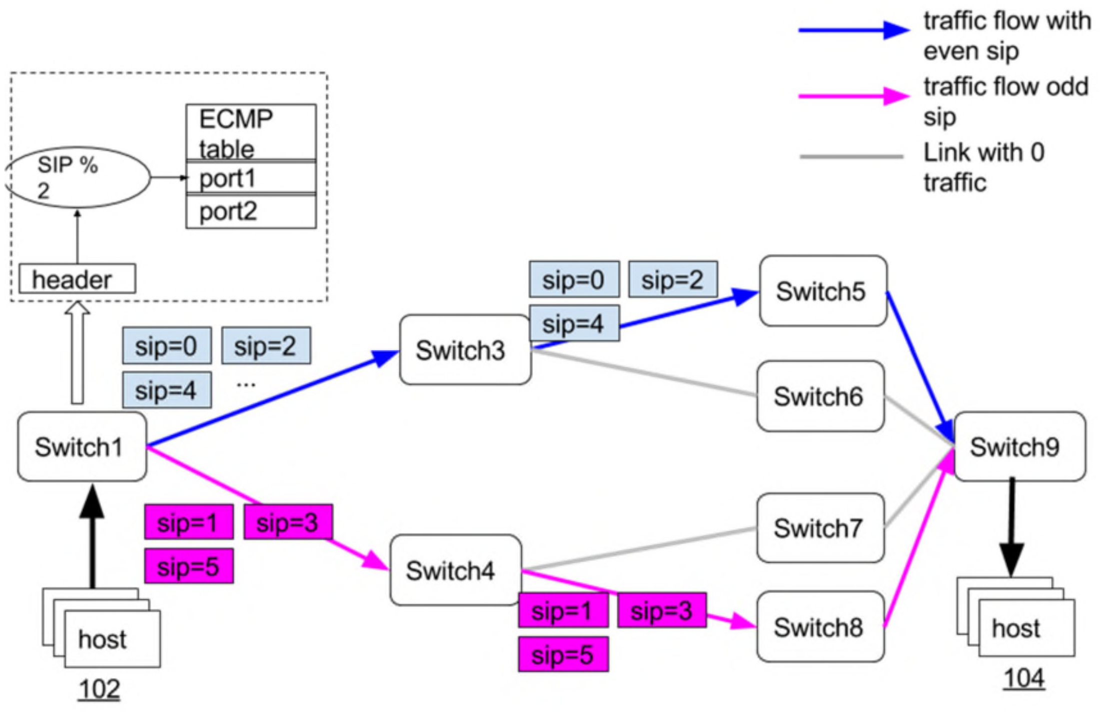

Fig. 1: Illustrating traffic imbalance in networks

In Fig. 1, hosts (e.g., servers) 102 are coupled to switch 1, and hosts 104 are coupled to switch 9. Switches 1, 3, and 4 each have two outgoing links. Packets that travel between hosts 102 and hosts 104 can be routed across multiple paths, e.g., 102-switches(1-3-6-9)-104, 102-switches(1-4-8-9)-104, etc. The hash function at each switch is, illustratively, the parity (even or odd) of the source internet protocol ID (SIP). For example, if the SIP of a packet is even (SIP modulo two equals zero) then the packet is routed via the upper outgoing link of the switch (illustrated in blue). If the SIP of a packet is odd (SIP modulo two equals unity) then that packet is routed via the lower outgoing link of the switch (illustrated in pink). With such a hash function that is identical at all switches, packets with even SIP will travel along the path 102-switches(1-3-5-9)-104, and packets with odd SIP will travel along the path 102-switches(1-4-8-9)-104. In such a configuration, the switch 3-6 and the switch 4-7 links are completely unutilized, whereas the switch 3-5 and switch 4-8 links carry a disproportionately high amount of traffic.

DESCRIPTION

Techniques of this disclosure address the problem of traffic imbalance in switching networks by using the VLAN (virtual local area network) tag of a packet as one of the inputs to a hash function, and by shuffling (e.g., randomly) the VLAN tag ("VLAN_ID") of the packet at switches along a path. The shuffling of VLAN tags at a switch is done in compliance with ACL (access control list) rules. In this manner, traffic load balance is achieved even when a limited number of hash functions are available.

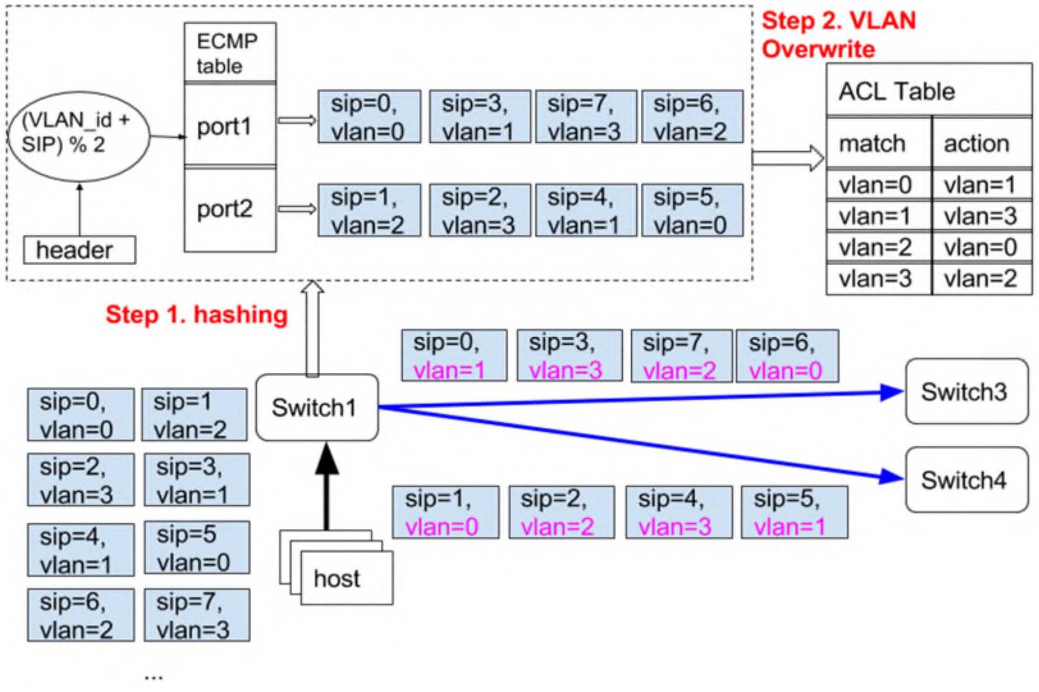

Fig. 2: Achieving traffic balance by using VLAN tag as an input to the hash function, and by shuffling the VLAN tag of a packet.

Fig. 2 illustrates load balancing within a network using techniques of this disclosure. At a first step ("step 1: hashing"), a switch, e.g., switch 1, routes a packet based on the parity of VLAN_ID+SIP. For example, if VLAN_ID+SIP is even (VLAN_ID+SIP modulo two equals zero), then the packet is assigned for routing through the upper outgoing link of switch 1 of Fig. 1. If the VLAN_ID+SIP is odd (VLAN_ID+SIP modulo two equals unity), then the packet is assigned for routing through the lower outgoing link of switch 1. Next ("step 2: VLAN overwrite"), the switch, e.g., switch 1, overwrites, prior to routing, the VLAN_ID of each packet using ACL rules. The mappings between old and new (overwritten) VLAN IDs are randomized on each switch, such that different switches do not use the same mapping. After such overwriting, the VLAN_ID+SIP of half of the packets assigned to the upper outgoing link is odd, while the VLAN_ID+SIP of half the packets assigned to the lower outgoing link is even.

In this manner, half the packets received by switch 3, which is connected to the upper outgoing link of switch 1, have an even value for VLAN_ID+SIP, while the other half have an odd value for VLAN_ID+SIP. Similarly, half the packets received by switch 4, which is connected to the lower outgoing link of switch 1, have an even value for VLAN_ID+SIP, while the other half have an odd value for VLAN_ID+SIP. Such rewriting of the VLAN_ID permits downstream switches, e.g., switch 3, switch 4, to evenly split their outgoing traffic between their respective two outgoing links, even when each downstream switch uses the same hash function, e.g., the parity of VLAN_ID+SIP.

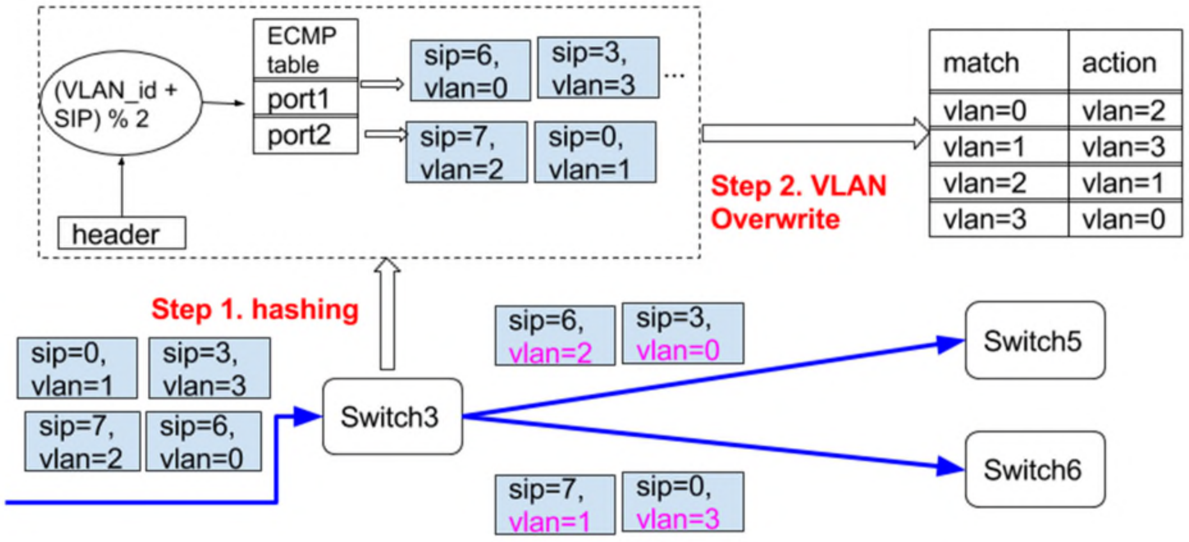

Fig. 3: Hashing behavior at downstream switch 3

The routing behavior and balanced traffic outflow at a downstream switch, e.g., switch 3, is illustrated in Fig. 3. As described above with reference to switch 1, in a first step (“step 1: hashing”), switch 3 uses as hash function the parity of VLAN_ID+SIP to determine the egress port of a packet. Since half of packets incoming to switch 3 have even VLAN_ID+SIP parity, load is evenly balanced across the two outgoing links of switch 3. Next (“step 2: VLAN overwrite”), the VLAN_ID of each packet is overwritten, prior to routing and in compliance with ACL rules. Prior to such overwriting, the VLAN_ID+SIP of packets assigned to the upper outgoing link of switch 3 have even VLAN_ID+SIP. The VLAN_ID is overwritten at switch 3 such that half the packets leaving the upper outgoing link of switch 3 and arriving at switch 5 have an odd value for VLAN_ID+SIP. In this manner, load balancing is achieved at switch 5.

In order to maximally decorrelate the hashing among different switches, it is important that each switch map differently from an old VLAN_ID to a new VLAN_ID. One way to implement such a mapping is to use Knuth shuffle algorithm. An advantage of the Knuth shuffle algorithm is that it has a uniform probability of generating each possible map. The probability of a map as generated by the Knuth shuffle algorithm equals the reciprocal of the factorial of the number of VLAN tags.

The VLAN tag is typically a 12-bit field, e.g., that supports 4096 different VLAN_IDs. To allow the use of VLAN tags for multiple applications, the range of VLAN_IDs used for hashing can be limited, e.g., to 0-511. This can be achieved by programming 512 rules to map a VLAN_ID between 0 and 511 to a new value within the same range.

CONCLUSION

Hashing functions are typically provided by a network switch to enable the routing of network packets in a load-balanced manner. Hashing functions are often reused among switches within a network. Such reuse can lead to imbalanced traffic distribution within the network. Techniques of this disclosure achieve load-balanced traffic distribution even with a limited number of hash functions. The techniques utilize VLAN tags as additional inputs to the hash functions. Further, the VLAN tag of a packet is randomly shuffled as it passes through a switch, permitting the next switch downstream to operate in a load-balanced manner.