MLV: A Multi-dimension Routing Information Exchange Mechanism for Inter-domain SDN

Ze Chen, Jun Bi, Yonghong Fu, Yangyang Wang, Anmin Xu

Tsinghua University
Nov, 2015
1. Motivation

The inter-domain networks use Border Gateway Protocol (BGP) to exchange the inter-domain routing information between Autonomous Systems (ASes). Though the inter-domain traffic forwarding depends on BGP during the past years, the Internet is facing the requirements of more and more novel inter-domain routing applications, such as multi-path routing, traffic engineering and multi-dimension routing.
1. Motivation

Short comings of BGP:

- **Destination IP addressed inter-domain routing**
  
  The routing policies of BGP are depend on the destination IP prefixes. However, novel network applications may require to distinguish routing by source addresses or some other fields in packet header to optimize the inter-domain routing path.

- **Single path inter-domain routing**
  
  When a BGP router receives a prefix from multiple paths, it chooses a best route and announces it to its other neighbors. It is not easy for BGP to provide multi-path inter-domain routing.

- **Inflexible inter-domain routing programmability**
  
  Mostly, BGP chooses the shortest AS path for inter-domain routing, customers can not customized an inter-domain routing path.
In this paper, we extend the advantage of SDN to an inter-domain network federation to improve the Internet routing flexibility. To achieve this goal, we propose a Multi-dimension Link Vector network view exchange mechanism (MLV) to exchange the flexible inter-domain routing information across multiple SDN domains. The architecture of MLV is shown in Figure1.

Figure1: MLV architecture
2. MLV Overview

• Three Features of MLV

Fine granularity. Unlike BGP’s destination IP based reachability information propagation, MLV announces multi-dimension prefixes, such as source and/or destination IP addresses, source and/or destination TCP/UDP port numbers, protocol number, etc.

Novel Routing Propagation Model. To support path diversity, we propose a novel routing propagation model named link vector. It delivers link related fine-grained inter-domain prefix information between ASes to construct fine-grained inter-domain network view.

Traditional AS Commercial Relationships. In this paper, the routing information propagation of MLV follows AS commercial relationships, including customer-provider, peer-peer and sibling-sibling.
2. MLV Overview

- Contributions of MLV

1. Multi-dimension IP addressed inter-domain routing
2. Multi-path inter-domain routing. MLV announces the fine-grained prefixes to its neighbors from all inter-domain links according to the commercial relationship.
3. Each AS can obtain a fine-grained inter-domain network view. Applications can customize a path for a specific fine-grained IP prefix.
3. MLV Mechanism

We introduce the MLV mechanism from five aspects:

- Virtual Network View Abstraction
- MLV Algorithm
- Network View Storage
- How to use the Network View
- MLV Exchange Example
3.1 Virtual Network View Abstraction

Taking into account the privacy of the information within the AS, we construct the intra-domain virtual network view with all edge switches of each AS.

The whole inter-domain network view is composed of intra-domain virtual links and inter-domain physical links.
3.2 MLV Algorithm

- **Initialize and Periodic Updates**

  In the network initialization phase, each AS constructs link vector messages and sends them to its neighbors according to inter-domain commercial relationships (provider to customer, customer to provider or peer to peer).

  The information encapsulated in a link vector message includes a intra-domain prefix set and the corresponding link list that allows the prefix to pass through. \{prefix, link1,link2,linkN\}.

  Meanwhile, each link in the link vector message has a periodic version number for updating. When link change happens, AS updates the version number of the related link in its domain, and sends the new information to its neighbors.
3.2 MLV Algorithm

- Inter-domain routing policy Exchange Algorithm

\[
\text{recvAS} \xrightarrow{\text{link vector message}} \text{localAS} \xrightarrow{\text{link vector message}} \text{toAS}
\]

\text{localAS} \text{ represents an AS itself}

\text{recvAS} \text{ denotes its neighbor who sends the link vector message to it}

\text{toAS} \text{ denotes the other neighbors it will send the new constructed link vector message to.}
3.2 MLV Algorithm

- **Inter-domain routing policy Exchange Algorithm**

When an AS receives a link vector message from its neighbor, it checks whether the exchange condition is satisfied among the recvAS, localAS, and toAS.

If the relationship among recvAS, localAS, and toAS meet one of the following situations, the exchange condition is satisfied. (Gao et al. [9])

1) The relationship between localAS and recvAS is provider to customer, and the relationship between localAS and sendAS is provider to customer or peer to peer.

2) The relationship between localAS and recvAS is peer to peer, and the relationship between localAS and toAS is provider to customer.

3) The relationship between localAS and recvAS is customer to provider, and the relationship between localAS and sendAS is provider to customer.
3.2 MLV Algorithm

- **Inter-domain routing policy Exchange Algorithm**

When the *localAS* receives a link vector message from its neighbor *recvAS*, it checks the intra-domain virtual links and the inter-domain links to the *toAS* which allow the received prefix to pass through, and adds these links together with the received link list to construct a new link list.

To reduce inter-domain exchange traffic, if any link in the new link list together with the prefix is already sent to the *toAS*, it will filter the link out of the new link list.
In MLV, when an AS receives link vector messages, they store the prefixes and link lists in its controller’s NoSQL database. To save storage space, each link in the network view has a prefix list, and all the prefixes that allow to pass through the link are recorded. The link field includes the information concerning the link, such as source AS, source switch, source port, destination AS, destination switch, destination port, version number, link utility and bandwidth, etc. The storage structure of the link vector information is shown in Figure 2.

Figure 2: Link Vector storage structure
3.4 How to use the Network View

By the received fine-grained prefixes and link lists information, each AS can get a specific inter-domain network view for a prefix by creating the index for the prefix, and provide the network view to the upper routing applications. The network view information for a specific prefix is shown in Figure 3.

Figure 3: Network view for a specific prefix
3.5 MLV Exchange Example

AS1 announces a prefix set P to all its neighbors in the topology based on the commercial relationship.

1) \( AS1 \rightarrow AS2 : \{P, \text{Link1}\} \);
   \( AS1 \rightarrow AS3 : \{P, \text{Link2}\} \);

2) \( AS2 \rightarrow AS3 : \{P, \text{Link1}, \text{VL21}, \text{Link3}\} \);
   \( AS2 \rightarrow AS4 : \{P, \text{Link1}, \text{VL22}, \text{Link4}\} \);

3) At the same time AS3 received a message from AS1
   \( AS3 \rightarrow AS4 : \{P, \text{Link2}, \text{VL31}, \text{Link5}\} \);

Later, AS3 receives another message sent from AS2.

   \( AS3 \rightarrow AS4 : \{P, \text{Link2}, \text{VL31}, \text{Link5}\} \);

4) Finally, AS4 receives an inter-domain network view with all the links that can pass through the prefix P.

"Figure 4: Example Topology"
4. Evaluation

• Simulation

In this part, we compare MLV with a Hop-by-Hop exchange mechanism to show the advantages and effects of MLV.

The difference between MLV and the Hop-by-Hop mechanism is that MLV exchanges inter-domain network view while Hop-by-Hop exchanges inter-domain paths. In Hop-by-Hop, when an AS receives a prefix from multiple paths, it stores the prefix and all paths.
4.1 Simulation

We write two simulation programs of Hop-by-Hop and the MLV exchange mechanism, and compare the exchange traffic and the storage space between them.

We use two kinds of topologies to compare the two exchange mechanisms: a basic tree topology and CAIDA data-sets topology.

The following two tests are proceeded under a basic tree topology, shown in Figure 6 (Gao [9] points out that the AS topology is a hierarchical structure).

Figure 6: Tree Topology
4.1 Simulation

- **Exchange Traffic test**

In the first exchange traffic test, each AS sends messages to its neighbors with prefix and link list according to the commercial relationship.

To simplify the comparison, we set minimum size for the prefix field (4 bytes) and link field (32 bytes, only including the basic information on a link: source AS, source switch, source port, destination AS, destination switch, destination port, version).

We range the number of trees, and we static the exchange traffic of all ASes. From Figure 7, we can observe that with the increase of tree numbers, the exchange traffic of the MLV exchange mechanism is much lower than the Hop-by-Hop exchange mechanism.

![Figure 7: Comparison of the exchange traffic between the Hop-by-Hop and the MLV exchange mechanism.](image-url)
4.1 Simulation

- **Storage Space Test**

In the second storage space test, all the assumptions are same with the first test. We compare the storage space between the two kinds of exchange mechanisms. From Figure 8, we can see that with the increase of tree numbers, the storage space of the MLV mechanism is far less than the Hop-by-Hop exchange mechanism.

![Figure 8: Comparison of the storage space between the Hop-by-Hop and the MLV exchange mechanism.](image)
4.1 Simulation

- Simulation based on CAIDA Data-Sets

Further, we compare the two exchange mechanisms under the real network topology. We use the Internet AS topology data-sets collected at October, 2014 by CAIDA. In the topology data-sets, we extract a tree that has 1155 ASes from the whole topology, and we get the simulation results.

The exchange traffic values of Hop-by-Hop and MLV exchange mechanism are \(2.4575 \times 10^9\) and 852055292 bytes. The storage space values of Hop-by-Hop and MLV exchange mechanism are \(2.4577 \times 10^9\) and 222976796 bytes.

The exchange traffic of the MLV accounts for 34.6% of the Hop-by-Hop. Meanwhile, the storage space of MLV accounts only for 9% of the Hop-by-Hop exchange mechanism. From the above simulation tests, we can observe that the MLV exchange mechanism effectively reduces the huge fine-grained exchange traffic and storage space.
4.2 Experiment

• Testbed

We build an inter-domain SDN testbed across multi-counties under the collaboration among education research networks in the USA, China, and Europe. The testbed topology is shown in Figure 9. The testbed has deployed hardware OpenFlow[13] switches from Pica8, Dell, etc, and software switches running Open vSwitch.

Figure 9: The structure of our inter-domain SDN testbed.
We evaluate the performance of the MLV based on the testbed above. We adopt four SDN domains: CST- NET, Internet2, CERNET and BUPT.

The AS commercial relationships between the four SDN domains are set as follows. The relation between CSTNET and Internet2 is provider to customer; The relation between CSTNET and CERNET is provider to customer; The relation between Internet2 and CERNET is provider to customer; The relation between CERNET and BUPT is provider to customer.

In this test, we range the number of updates of each SDN domain from 100 to 1000 times, the convergence time of MLV increases from 50s to 450s. The convergence time of MLV is shown in Figure 10.
5. Discussion

• 5.1 Scalability

Though the fine-grained network view exchange mechanism of MLV may bring scalability problem, we consider it is acceptable to a network federation. In this part, we analyze the topology data-set of CAIDA which is collected at October 2014. In the dataset, there are 46120 ASes in the whole topology. We calculate the cumulative distribution of ASes’ degree, shown in Figure 11.

33.23% ASes’ degree is equal to 1;

68.66% ASes’ degree is less than and equal to 2;

79.59% ASes’ degree is less than and equal to 3;

84.29% ASes’ degree is less than and equal to 4;

90.65% ASes’ degree is less than and equal to 8. The average degree of each AS in the data-set is 7.
We analyze the Total Storage Space (TSS) of MLV in a network federation.

Assume there are $N$ ASes in a network federation; the average degree of each AS is $d$; the average virtual links number in each AS is $c$. Then there is $(d*N)/2 + c*N$ links in a network federation.

The MLV mechanism uses the structure <link, prefix1, prefix2,…, prefixN> (shown in Figure 2) to store the inter-domain network view. As the size of the link structure is much less than the total size of the related inter-domain prefixes which are permitted to go through it, we omit this part.

Then TSS is equal to the total links number multiply by the average prefixes number related to a given link.

$$TSS = \left(\frac{d*N}{2} + c*N\right) \times \text{Sizeof(AvgPrefixNum)} = O(N) \times \text{Sizeof(AvgPrefixNum)}.$$ 

In TSS, the average prefixes number related with a given link is much less than all the inter-domain prefixes number stored in a traditional BGP router. Meanwhile, we do not install all the inter-domain fine-grained forwarding policies into the switches in the data plane, they are stored in the memory of controllers of the network federation.
5.2 Convergence

A link vector message of MLV is divided into many prefix-link pairs. When an AS in the network federation gets a link-prefix pair out of a received link vector message, the controller of the AS searches the link-prefix pair information in its storage.

If the version of the pair is same with the version of the corresponding existed pair stored in its storage, it means that the controller has already received it and broadcasted it to its neighbors, so the AS will not broadcast it again. Then we know, each AS only broadcasts the same version link-prefix pair information once. Thus, MLV can converge.
In this paper, we propose an effective multi-dimension link vector network view exchange mechanism to exchange the huge fine-grained inter-domain network view and enable flexible inter-domain routing innovation. Meanwhile, we analyzed the performance of MLV with real Internet AS topology. Finally, we implemented a prototype of MLV to verify its feasibility and effectivity, and tested it on an inter-domain SDN testbed interconnecting multiple SDN networks.
Thank You!

Q&A