IVeri: A Scaling Privacy-Preserving Interdomain Configuration Verification Tool via Secure Multi-Party Computation

Appendix

A. An Example of Converting a BGP Policy to a Vector of Integers

In IVeri, the content involved in converting a BGP policy to a vector of integers can be divided into three parts: the information related to the AS's peer relationships, import policies, and export policies. Next, I will illustrate our conversion process with a specific example. We use the Cisco format AS configuration, which mainly includes the following three parts: information related to the topology with other ASes, routing relationships for each edge (e.g., customer-provider relationship, peer-peer relationship, sibling relationship, etc.), and the route maps corresponding to those routing relationships. The example configuration is shown in figure 2.

```
174 4230 58453 701 7018 7713 9318
3 0 1 2 4 5 6
0 0 0 1 1 1 0
0 0 0 167772178 167772194 167772226 0
0 0 0 174 174 174 0
0 0 0 701 7018 7713 0
0 0 0 167772176 167772192 167772224 0
```

Fig. 1: The AS topology relationships after conversion to vectors.

1) Conversion of AS Topology and Edge Relationships to Vectors

Our conversion algorithm begins by processing and converting the information related to the topology with other ASes. The AS topology relationship information from the example configuration is converted and shown in Figure 1. The first step in this conversion is a discretization process, where each AS number is mapped to a unique index. For instance, AS174 is mapped to index 3, and AS7018 to index 4. This guarantees that each AS involved in the verification process receives a unique index. To store this mapping relationship, we use two vectors, which correspond to lines 1 and 2 of the converted vector file.

Following this, we use a vector to represent the edge relationships of each AS. The index of a specific AS is set to either 0 or 1 depending on whether the configuration file indicates a connection to the corresponding AS. For example, for AS174, if it has connections to the nodes at indices 4, 5, and 6, the corresponding indices will be set to 1. This is reflected in line 3 of the converted vector file.

Additionally, line 4 contains the local IP address associated with the link's interface, while line 5 represents the local ASNumber. Lastly, lines 6 and 7 hold the interface information for the other endpoint of the link, including the remote ASNumber and the IP address associated with that interface.

2) Conversion of Import Policy to Vectors

Since each pair of topology connections in the routing configuration includes a routing relationship between the two nodes, such as customer-provider relationships, peer-peer relationships, etc., the import policy is to convert the corresponding route map into a vector based on the relationship of that connection. First, we determine whether an edge relationship exists by checking the third line of the conversion in A1. If it exists, we convert the corresponding route map into a vector. We have converted the edge between AS174 and AS701, which is shown in Figure 2, into a vector representation, as illustrated in Figure 3a.

Our handling of the import policy is divided into two categories: one is for policies specific to certain prefixes, and the other is for policies applicable to all prefixes. In the example, we demonstrate the policy for all prefixes. To ensure that the number of import and export policies cannot be deduced by the amount of data transferred, we use a fixed number of policy entries (for the sake of demonstration, the example uses five entries) to obscure the actual number of policies. For example, if there is only one real entry, the remaining four entries will be filled with default content.

Firstly, we handle the import policy that applies to specific prefixes, with each column representing one policy. Line 1 specifies the prefix to which the policy applies, and Line 2 indicates the subnet mask for that prefix. These two lines together define the scope of the policy. Line 3 represents the action to be taken, determining whether packets for the prefix are permitted or denied. Lines 4 and 5 pertain to the community associated with the policy. Line 4 indicates the type of the community, while Line 5 specifies the global identifier associated with that community. These two lines define the characteristics and behavior of the community in relation to the policy.

For policies that apply to all prefixes (this part should not

```
interface Ge0/0
 ip address 10.0.0.19 255.255.255.252
 description "To␣AS701"
 speed auto
 duplex auto

interface Ge1/0
 ip address 10.0.0.35 255.255.255.252
 description "To␣AS7018"
 speed auto
 duplex auto

interface Ge2/0
 ip address 10.0.0.67 255.255.255.252
 description "To␣AS7713"
 speed auto
 duplex auto

interface Ge7/0
 ip address 10.50.2.1 255.255.255.0
 description "Host␣Network"
 speed auto
 duplex auto

neighbor 10.0.0.18 remote-as 701
neighbor 10.0.0.18 description "To␣AS701"
neighbor 10.0.0.18 route-map INPROV in
neighbor 10.0.0.18 route-map OUTPROV out

neighbor 10.0.0.34 remote-as 7018
neighbor 10.0.0.34 description "To␣AS7018"
neighbor 10.0.0.34 route-map INPEER in
neighbor 10.0.0.34 route-map OUTPEER out

neighbor 10.0.0.66 remote-as 7713
neighbor 10.0.0.66 description "To␣AS7713"
neighbor 10.0.0.66 route-map INCUST in
neighbor 10.0.0.66 route-map OUTCUST out

route-map INPEER permit 10
 set local-preference 80
 set community 65000:1

route-map OUTPEER deny 10
 match community 10

route-map OUTPEER permit 20

route-map INPROV permit 10
 set local-preference 60
 set community 65000:2

route-map OUTPROV deny 10
 match community 10

route-map OUTPROV permit 20

route-map INCUST permit 10
 set local-preference 100

route-map OUTCUST permit 10
```

Fig. 2: AS Configuration in Cisco Format.

be translated, as instructed), Line 6 represents a flag indicating whether this column contains a policy for all prefixes. If so, the value is set to 1; otherwise, it is set to 0. Line 7 shows whether the policy applies to the next-hop. If it does, the next-hop's IP prefix is provided (represented as a 64-bit integer). Line 8 specifies the LocalPreference set by the policy, while Line 9 represents the metric attribute of the BGP route set by the policy. Finally, the last two lines correspond to whether the policy matches a community. If so, they indicate the type and global identifier of the community.

```
0 0 0 0 0
0 0 0 0 0

-1 -1 -1 -1 -1

-1 -1 -1 -1 -1

-1 -1 -1 -1 -1

1 0 0 0 0
0 0 0 0 0
60 100 100 100 100
-1 -1 -1 -1 -1
65000 0 0 0 0
2 0 0 0 0
```

(a) Import Policy.

```
0 0 0 0 0
0 0 0 0 0


-1 -1 -1 -1 -1

-1 -1 -1 -1 -1

-1 -1 -1 -1 -1

1 0 0 0 0
0 0 0 0 0
-1 -1 -1 -1 -1
-1 -1 -1 -1 -1
1 0 0 0 0
```

(b) Export Policy.

Fig. 3: The Import Policy and Export Policy after Conversion to Vectors.

3) Conversion of Export Policy to Vectors

The parsing and conversion logic of the export policy is similar to that of the import policy, so it won't be repeated here.