Network Topology Discovery through IS-IS

Zhenyuan Bo, Xiao Wang
Electrical and Computer Engineering Dept.
University of Toronto
zhenyuan.bo@mail.utoronto.ca
fx.wang@mail.utoronto.ca

***Abstract*—Telecom Networks industry has undergone massive innovation and revolution over the decade. This era brings the creations of many advanced equipment as well as software infrastructure. With such a high volume and complexity of networks, it is beneficial to have an appropriate tool to discover the network topology and to analyze for better understanding of the network. In this paper, we will focus on topology discovery through IS-IS. In the end, the users can visualize the physical and visual network elements as well as how they are connected on a map.**

Keywords—Topology Discovery; Network Management; IS-IS protocol.

I. INTRODUCTION

The Internet network scale is growing exponentially nowadays. With more and more users and applications creating traffic over multiple terabits or even pita-bits, the network is at a huge risk and service providers have issues with this ever-growing network. Issues include the lack of complete network connectivity, inaccurate and out of date network diagrams, unavailable information of connections on different vendors and layers. However, if gaining the insight of network topology, which is the network connectivity, we can manage the network resources proactively, and conduct traffic flow analysis and contingency analysis. Consequently, it will further save up bandwidth and budget.

Prior to discover the topology, we need to know the routing protocol that the network is compatible with. IS-IS is short for Intermediate System-to-Intermediate System (IS-IS) [1]. It is a link-state routing protocol and works similar to OSPF (Open Shortest Path First). There are many topology discovery research studies available for OSPF publicly but not IS-IS. Thus this project is based on the IS-IS protocol.

II. BACKGROUND

A. Topology Discovery

Topology discovery is a functionality that returns a map of network elements and their inter-connections. Given the ability to visualize the network layout, service providers can mitigate network problems by isolating the relevant network resources. It is similar to utilities when they face power outage. If you know the layout of the outage area, you can isolate and troubleshoot the relevant equipment quickly.

There are several approaches to achieve network topology discovery. In the past, network pioneers used network tools such as Ping, Traceroute and Simple Network Management Protocol (SNMP) [2]. Ping function is to check if the destination is available while Traceroute returns the traversing path and the trip time. By pinging/tracerouting network elements the mapping can be generated. But the shortage is obvious. By running either of the commands, extra network load gets added into the network. Later SNMP became the optimal choice [3]. SNMP is one of the most popular and powerful tools. Tens of millions of SNMP probes are deployed across the world to extract network traffic. It can provide you data information quickly. However, SNMP cannot return real-time data.

Nowadays the state-of-the-art is to implement topology discovery with OSPF protocol. It is a passive monitoring approach. It solves the issues of extra load and real time limitation. The idea is simple. Routers are administered by the routing protocol. Since OSPF is a link-state routing protocol which means each router has the mapping within its own area. In that case, a complete routing table is stored in every router in the network. If any of the routing table can be acquired, the whole network can be visualized.

B. IS-IS protocol

IS-IS (Intermediate System-to-Intermediate System Protocol) is a link state routing protocol. IS-IS is an OSI routing and Interior Gateway Protocol (IGP). The protocol is similar to OSPF, which uses Dijkstra algorithm to build a IS-IS database and find the best next-hop [4]. By acquiring the routing table of anyone node, we can observe the whole network connectivity.

IS-IS is a 2-level hierarchy protocol. It contains Level-1, the area networks and Level-2, the backbone network. As a result, different types of routers are deployed. There are Level 1 router, Level 2 router and Level 1/Level 2 router. Level 1 routers know the neighbors and the routing table within the area. Level 2 routers has the database for intra-area routing table as well as inter-area topology. Level 1/Level 2 routers communicate between Level 1 and Level 2 networks. They have both the backbone network and the area network routing information.

IS-IS used SNAP(Subnet Point of Attachment) for

OSPF is well-known and is used more widely comparing to IS-IS. However, IS-IS has some differences and advantages in certain areas. First, IS-IS headers encapsulate on the second layer of OSI model, which is the data link layer. [5] On the other hand, OSPF headers depend on IP-network packets. This makes IS-IS more adaptive with non-IP traffic. Furthermore, with the adoption of IPv6, OSPFv2 and OSPFv3 are both required to support IPv4 and IPv6. That means, one network with two similar configurations co-exist with different versions. This doubles the complexity and workloads. As a result, IS-IS will apparently re-converge much faster and perform more reliable than OSPF. Also, IS-IS is more scalable and simpler to operate. These reasons make IS-IS the IGP choice for internet service providers. Therefore, as a final project for service provider networks course, IS-IS is chosen as the routing protocol for the experiment network.

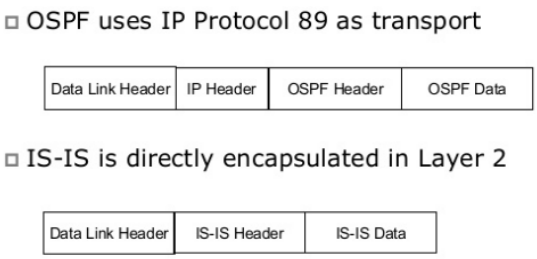


Figure 1. IS-IS and OSPF header [5]

C. Model

In this paper, we used Abilene network as our simulation network model. Abilene network was a high-performance backbone network created by Interenet2 in the late 1990s. [6] It was deployed across the U.S. and contained 12 nodes.

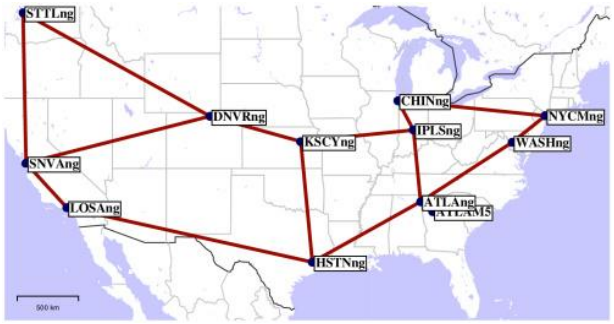


Figure 2. Abilene Network

Abilene network is recommended by the course instructor and is provided with large traffic dataset.

D. Visuilazation Tool

During experiment, Cytoscape is utilized as the topology visualization tool. Cytospace is an open source software platform for visualizing molecular interaction networks. [7] It takes a file with graphml format as an input and produces a network topology diagram.

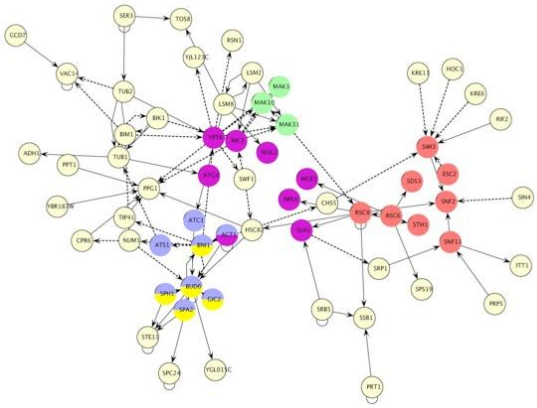


Figure 3. Sample Cytoscape product

III. ALOGRITHM

IS-IS is a link state protocol that utilizes Dijkstra algorithm to find out the best path within the network. In theory, one node can provide us the complete connectivity mapping because it has the routing table for the whole network. The key is to find the command to show the routing table.

In the experiment, GNS3 command “Show isis topology” was used [8]. This command provides information about the link connection among nodes in the network. However, some connection needs more deeper analysis to prove their validity especially when the metric value is very large. An example return of the command is shown below. It returns information in five fields: System ID, Metric, Next-Hop, Interface and SNPA. The System ID simply lists a number of nodes within the area. The metric value means the distance between the distant nodes and the current node. Index 10 means direct connection, index 20 means one hop away and index 30 is 2 hops away. The Next-hop field is the first node it should go from the accessing node in order to reach the corresponding node. The interface field shows the connecting interface port of the corresponding router. The SNPA (Subnetwork Point of Attachment) identifies a point at which a device connects to a network. It is typically the layer 2 address e.g. MAC address.

```
IS-IS paths to level-2 routers
System Id           Metric     Next-Hop            Interface   SNPA
NYCMng              20         WASHng              Fa2/0       cc10.349b.0010
ATLAng              --
ATLA-M5             10         ATLA-M5             Fa3/0       cc0c.3116.0000
KSCYng              20         HSTNng              Fa0/0       cc06.3110.0020
                               IPLSng              Fa1/0       cc07.3111.0020
HSTNng              10         HSTNng              Fa0/0       cc06.3110.0020
CHINng              20         IPLSng              Fa1/0       cc07.3111.0020
IPLSng              10         IPLSng              Fa1/0       cc07.3111.0020
WASHng              10         WASHng              Fa2/0       cc10.349b.0010
```

Figure 4. “show isis topology” sample output

From the previous command, it is clear that System ID, Metric and Next-Hop are the three key fields concern us. We can use these 3 sets of number to define each node and possible connectivity.

To verify the above network, we searched other GNS3 isis related commands. There is another command that could bring useful information. That is the “show isis rib”. RIB here stands for Routing Information Base. This command displays paths for all routes under one network. We can observe from below that it specifies every link, and its destination and interface port.

```
192.168.1.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
192.168.2.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.3.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.4.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
192.168.5.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.6.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.7.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
192.168.8.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
```

Figure 5. “show isis rib” sample output

IV. Experiment

The experiment can be divided into 4 parts. The first part was to build the Abilene network and implemented IS-IS protocol. Second part was to extracted data from one of the routers. The third part was to analyze and parse the data captured. The last part was to visualize the analyzed data through an open source platform

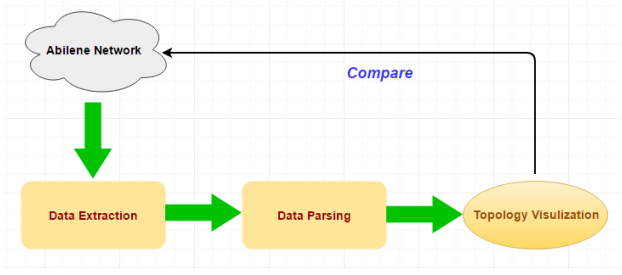


Figure 6. Experiment procedure

1.0 Network Set-up

To experiment, we used GNS3 to simulate the Abilene network. Twelve nodes were built and connected as the Abilene network.

Next, we incorporated the IS-IS protocol into all the nodes. The whole network will act as a level 2 backbone network. Basically, we added IP addresses for each router and configured it using IS-IS protocol. We also needed to set the Network System ID manually for each router.

Example configuration for a router

>>conf t

***enter the configuration terminal

>>int f1/0

***enter the configuration for fast Ethernet 1/0 interface

>>ip add 192.168.1.1 255.255.255.0

***set an IP address and subnet mask

>> ip router isis

**implement IS-IS protocol

>>no shut

***turn on the interface

>>int lo0

***configure the loopback

>>ip add 1.1.1.1 255.0.0.0

***add loopback address and subnet mask

>>ip router isis

**implement IS-IS protocol

>>exit

Example configuration for router

>>router isis

***enter router configuration

>>net 49.0001.1111.1111.1111.00

***Assign the NET system ID

>>is-type level-2

***set the router as level-2 i.e. backbone router

Table 1. Loopback address for each router

Node Name	Loopback Address
STTLng (R1)	1.1.1.1
SNVAng(R2)	2.2.2.2
LOSAng(R4)	4.4.4.4
DNVRng(R3)	3.3.3.3
KSCYng(R5)	6.6.6.6
HSTNng(R6)	5.5.5.5

IPLSng(R8)	8.8.8.8
CHINng(R7)	7.7.7.7
WASHng(R9)	15.15.15.15
NYCMng(R10)	10.10.10.10
ATLAng(R11)	9.9.9.9
ATLA-M5(R12)	12.12.12.12

Table 2. NET System ID for each router

Node Name	NET System ID	LEVEL ID
STTLng (R1)	49.0001.1111.1111.1111.00	L2
SNVAng(R2)	49.0001.2222.2222.2222.00	L2
LOSAng(R4)	49.0001.4444.4444.4444.00	L2
DNVRng(R3)	49.0001.3333.3333.3333.00	L2
KSCYng(R5)	49.0001.5555.5555.5555.00	L2
HSTNng(R6)	49.0001.6666.6666.6666.00	L2
IPLSng(R8)	49.0001.8888.8888.8888.00	L2
CHINng(R7)	49.0001.7777.7777.7777.00	L2
WASHng(R9)	49.0001.9999.9999.9999.00	L2
NYCMng(R10)	49.0001.1010.1010.1010.00	L2
ATLAng(R11)	49.0001.1011.1011.1011.00	L2
ATLA-M5(R12)	49.0001.1012.1012.1012.00	L2

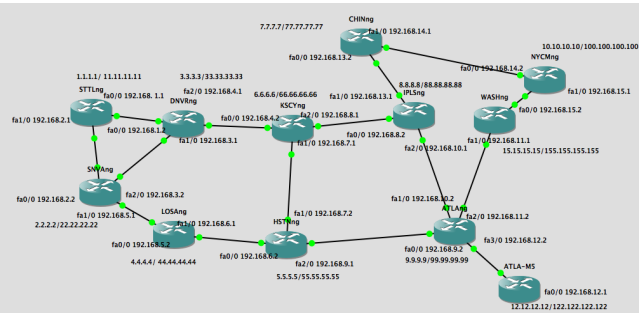


Figure 7. GNS3 network layout

2.0 Data Extraction

The IS-IS configuration was then complete and the network was fully operational. The next step should access a node and obtain a routing table for the whole network. In theory, every node can return a full map. Hence, we used an edge node, Seattle as our accessing router.

With the Seattle router, we used the command "show isis level-2 topology". The output showed all the routers within the network, the metrics, the next-hop, the interface and SNPA. We saved the output into a text file.

```
IS-IS paths to level-2 routers
System Id           Metric     Next-Hop             Interface   SNPA
NYCMng              60         DNVRng               Fa1/0       cc02.1cc0.0000
ATLAng              40         DNVRng               Fa1/0       cc02.1cc0.0000
ATLA-M5             50         DNVRng               Fa1/0       cc02.1cc0.0000
STTLng              --
SNVAng              10         SNVAng               Fa0/0       cc03.0a14.0000
DNVRng              10         DNVRng               Fa1/0       cc02.1cc0.0000
LOSAng              40         DNVRng               Fa1/0       cc02.1cc0.0000
KSCYng              20         DNVRng               Fa1/0       cc02.1cc0.0000
HSTNng              30         DNVRng               Fa1/0       cc02.1cc0.0000
CHINng              40         DNVRng               Fa1/0       cc02.1cc0.0000
IPLSng              30         DNVRng               Fa1/0       cc02.1cc0.0000
WASHng              50         DNVRng               Fa1/0       cc02.1cc0.0000
```

Figure 8. "show isis topology" output

The above table should be the routing table that could give us the visualization of the network. The interface and SNPA column here do not tell the connectivity of the network as far as we know. As a result, we used the metric (distance) and next-hop to determine the connectivity. However, after experiments, we found the information from these two columns was not sufficient to form the complete network due to the reason as it only tells us the immediate neighboring node. Thus, the connectivity complexity was not clear enough for some connections among distant router nodes.

The requirement of this project is to use only one node to visualize the whole network. Therefore, we investigated on more demands. The command "show isis rib" was used to complement the topology command. This command can display all the routes that are stored in the IP local Routing Information Base (RIB). The result essentially confirms the connectivity between any two routers in the network. This command clearly shows all the connectives both routers which well complements the topology command. Below is the screen for the rib command.

```
192.168.1.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
192.168.2.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.3.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.4.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
192.168.5.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.6.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.7.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
192.168.8.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
192.168.9.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
192.168.10.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
192.168.11.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 15.15.15.15, tag 0, LSP[16/94]
192.168.12.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 12.12.12.12, tag 0, LSP[24/93]
192.168.13.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 7.7.7.7, tag 0, LSP[15/98]
192.168.14.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 7.7.7.7, tag 0, LSP[15/98]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 10.10.10.10, tag 0, LSP[21/93]
192.168.15.0/24
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 15.15.15.15, tag 0, LSP[16/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 10.10.10.10, tag 0, LSP[21/93]
```

Figure 9. "show isis rib" output

3.0 Data Parsing

Data parsing and analysis is a crucial step in order to discover the topology. Given the previous step, we obtained two text files. Next, we transformed and analyzed the information from both files and then combine into one file, which contains information of the complete network topology. The flow chart below shows every detailed step.

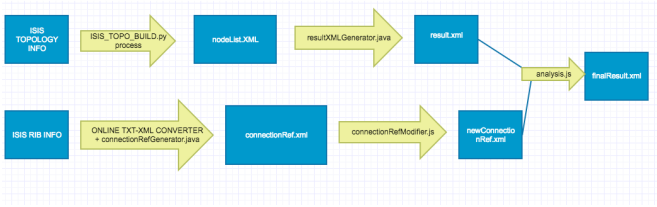


Figure 10. Data parsing flowchart

The two command files were processed separately at the beginning. The python script "ISIS_TOPO_BUILD.py" was created to extract data and convert it into XML files. Upon completion, several data elements were selected, including metrics and next hop as they indicated connectivity. The resulting files are "nodeList.xml"

```
<NodeList>
  <Node>
    <nodeId>NYCMng</nodeId>
    <Metric>60</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>ATLA-M5</nodeId>
    <Metric>50</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>SNVAng</nodeId>
    <Metric>10</Metric>
    <NextHop>SNVAng</NextHop>
  </Node>
```

Figure 11. Stage-1parsed topology command screenshot

It is clear that metric "10" suggests a direct connection while metric "20" suggests the destination is one hop away. Metric "30" or more node is a little tricky to process. Since we only know the first transit hop. So we bravely used lower level metric as the previous hop to the destination. For example, Nork York router has a metric of 60. We can conclude that all the routers with a metric of 50 are potential routers that connect to New York router. That is exactly the duty of the script "resultXMLGenerator.java". Up to now, the result.xml can figure out the first hop and the potential last hop.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<NodeList>
  <Node>
    <nodeId>NYCMng</nodeId>
    <Metric>60</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>ATLA-M5</fromId>
    <fromId>WASHng</fromId>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>HSTNng</fromId>
    <fromId>IPLSng</fromId>
  </Node>
```

Figure 12. Stage-2 parsed topology command screenshot

In fact, Washington and Chicago routers are the actual routers connect to New York router and Atlanda-M5 router is not although it has a metric of 50. Therefore, to verify the validity of potential last-hop routers, we analyzed the RIB command in order to complement the topology command.

Firstly, due to the complexity the output generated from the RIB command we used online txt to xml converter to speed up our experiment. Then to match all the router names, we wrote java script "connectionRefGenerator.java" to convert the loopback addresses to actual hostnames. Please note that a link has two connected routers and so we did not work on any link with only one connected router in its loopback address. That is why you will notice the first two entries in the picture below are still presented in number. Those are two entries connected to the Seattle node itself and so it cannot see through. For now, we will ignore Seattle node and it will be added later on.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<IpList>
  <IpRegion id="192.168.1.0/24">
    <fromId>3.3.3.3</fromId>
  </IpRegion>
  <IpRegion id="192.168.2.0/24">
    <fromId>2.2.2.2</fromId>
  </IpRegion>
  <IpRegion id="192.168.3.0/24">
    <fromId>DNVRng</fromId>
    <fromId>SNVAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.4.0/24">
    <fromId>DNVRng</fromId>
    <fromId>KSCYng</fromId>
```

Figure 13. Stage-1parsed RIB command screenshot

From above information, we used script "connection RefModifier.js" to furter clean up the data. We converted the data to a similar form to the toplogy xml file and represented the newConnectionRef.xml with all the connected router-pairs.

```
<NodeList>
  <Node>
    <nodeId>DNVRng</nodeId>
    <fromId>SNVAng</fromId>
  </Node>
  <Node>
    <nodeId>DNVRng</nodeId>
    <fromId>KSCYng</fromId>
  </Node>
  <Node>
    <nodeId>SNVAng</nodeId>
    <fromId>LOSAng</fromId>
  </Node>
  <Node>
    <nodeId>HSTNng</nodeId>
    <fromId>LOSAng</fromId>
  </Node>
```

Figure 14. Stage-2 parsed RIB command screenshot

That was the end of the process for the RIB command data. It was time to combine the processed topology command data and the processed RIB command data. It is clear that the topology command provides details of the layout and position of each router while RIB command specializes in understanding of their connections. Combing both files can get rid of some uncertainties and obtain a complete map of the network.

The "analyisi.js" plays the role of integrating the two files. In the topology data file, we have each node and its hostname, metric, next-hop and potential last-hops. In the other file, we have each node and its hostname and its connected hop. The connected hop is in fact the last hop. Therefore, the analysis code designs to match the node hostname in both files. If the node ID is matched, we will compare the last hop entry. In detail, if we find a match in last hop (fromid) , we add a new attribute "check " to "yes". If a connection (fromid) entry does not exist in the topology file, we add the entry to the topology file with the valid attribute "check". But we find a hop that exists in the topology file but not in RIB file, we are sure that hop is invalid and will be removed later.

We concluded above that the RIB file specializes in connectivity. So the connection judgements were based on the RIB file. We deleted all the last hop entries without the "check" attribute. In this way, we deleted all incorrect last hops and added missed ones. The output of the analysis code is final version that displays every router and its connectivity as the Abilene network.

```
<NodeList>
  <Node id="NYCMng">
    <fromId check="YES">WASHng</fromId>
    <fromId check="YES">CHINng</fromId>
  </Node>
  <Node id="ATLAng">
    <fromId check="YES">HSTNng</fromId>
    <fromId check="YES">IPLSng</fromId>
    <fromId check="YES">WASHng</fromId>
    <fromId check="YES">ATLA-M5</fromId>
  </Node>
  <Node id="ATLA-M5">
    <fromId check="YES">ATLAng</fromId>
  </Node>
  <Node id="SNVAng">
    <fromId>STTLng</fromId>
    <fromId check="YES">DNVRng</fromId>
    <fromId check="YES">LOSAng</fromId>
  </Node>
```

Figure 15. Combined command xml output

4.0 Topology Visualization

After a complete analysis for the data, we obtained an xml file that has everything needed to visualize the network. However, most visualization tools read the graphml format instead of regular xml. Therefore, we need to convert to graph xml format before virtualization.

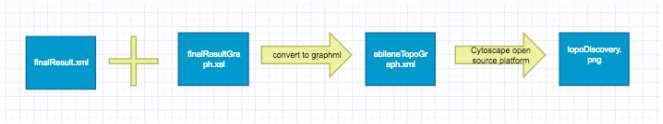


Figure 16. Visualization procedure

We used a translator that takes the xml file and a xsl file as inputs to transform to obtain the required graph xml file. The xsl file defines a set of rules and converts the xml file into a graphxml structure. The output returns a all router nodes and connections in between.

Finally, we imported the graph xml file to an open source virtualization tool, Cytoscape. The Cytoscape platform read the graph xml file and transformed into a diagram, which shows a network topology.

V. Results

This project was to discovery network topology through IS-IS protocol. Abliene network was modeled and extracted to recover the original topology. The final topology was visualized as below. This topology was exactly matched the actual Abilene network with some physical location offset. It is worth noting that the double connections between each router pairs represent the bi-directional property.

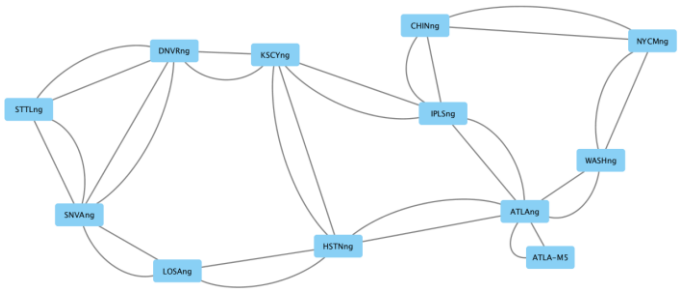


Figure 17. Final result: Topology visualization

Figure 18. Original Abilene network

VI. CONCLUSION

In this paper, we have experimented approaches to discover IS-IS implemented network topology with only one accessing node. The crucial part of topology discovery is to select the most valuable command and analyze the data to construct a complete network map. The result has demonstrated to be the same topology as the original Abilene network. The successful model of topology discovery can bring more insights for the network and consequently spare bandwidth, reduce outage time, save labor and budget.

Further improvements should be applied to make the discovery efficient and automatic given time. We could use one command instead of two to realize the topology. Assume one command being used, we can further reduce coding volume and complexity. Another direction is to combine the topology discovery with traffic analytics. Specifically, if we have a topology map, intuitively it will be much more effective to provide us real-time traffic by simply clicking on a router or a link.

REFERENCES

[1] NetProtocal Xpert, "IS-IS Protocol Introduction," SlideShare, Linkedin, Oct 03 2016. [Online]. Availble: *http://www.slideshare.net/NetworkersHome1/isis-protocol-introduction*

[2] X.Ma, G. Xia, "Autonomous Sytsme Network Topology Discovery Algorithm Based on OSPF Protocol," 3rd Intl. Conference on Material, Mechanical and Maufacturing Engineering (IC3ME), 2005.

[3] J. Wang, B. Chen, C. Tan, "Design and Implementation of Network Topology Discovery System Based on OSPF," Advanced Material Research, ISSN: 1662-8985, vol. 760-762, pp. 760-762, Sept 2013.

[4] F. Smith, "Migrating form OSPF to ISIS," MyNOG 3, Kuala Lumpur, Nov 29, 2013.

[5] M. Bhatia, "Why providers still prefer IS-IS over OSPF when designing large flat topologies," Routing Freak, Mar 5, 2011.[Online]. Available: *https://routingfreak.wordpress.com/2011/03/05/why-providers-still-prefer-is-is-over-ospf-when-designing-large-flat-topologies/*

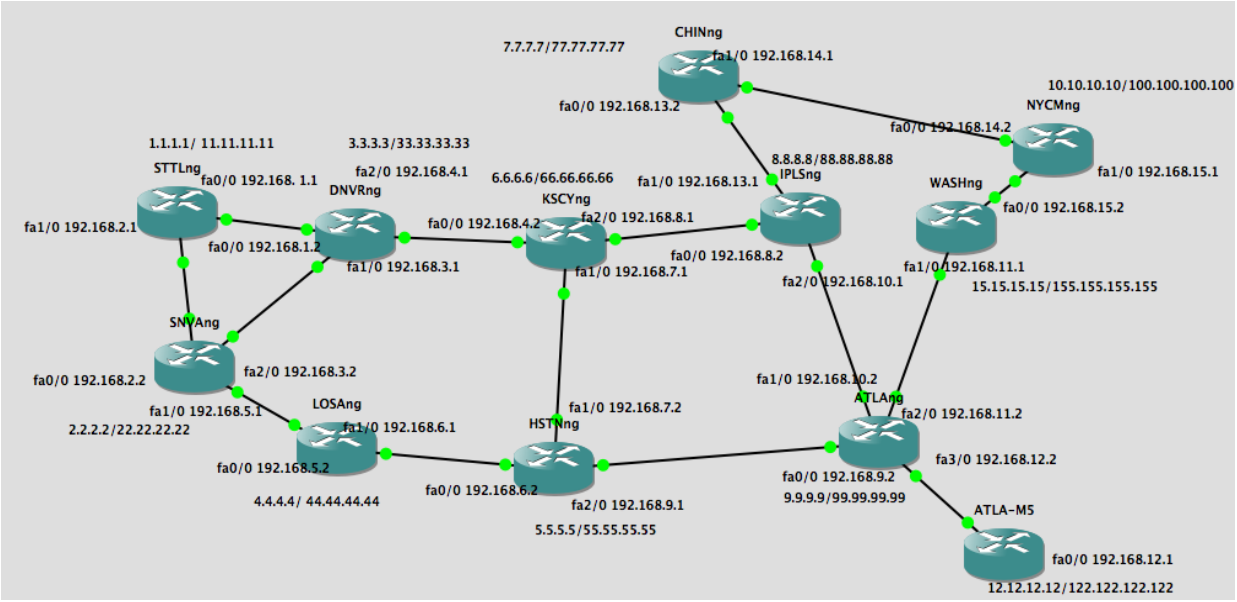
[6] R.H. Zakon, "Hobbes' Internet Timeline 10.1," Retrieved Dec 16, 2016

[7] P. Shannon, A. Markiel, O. Ozier, et al., "Cytoscape: A Software Environmnent for Integrated Models of Biomolecular Interaction Networks," Genome Research, 12(11): 2498-504, Nov 2003.

[8] "Cisco IOS IP Routing: ISIS Command Reference," Cisco Docs, Retrieved Dec 16, 2016. [Online]. Availble: *http://www.cisco.com/c/en/us/td/docs/ios/iproute_isis/command/reference/irs_book.html*

VII. APPENDIX

1. Abilene network Set-up on GNS3



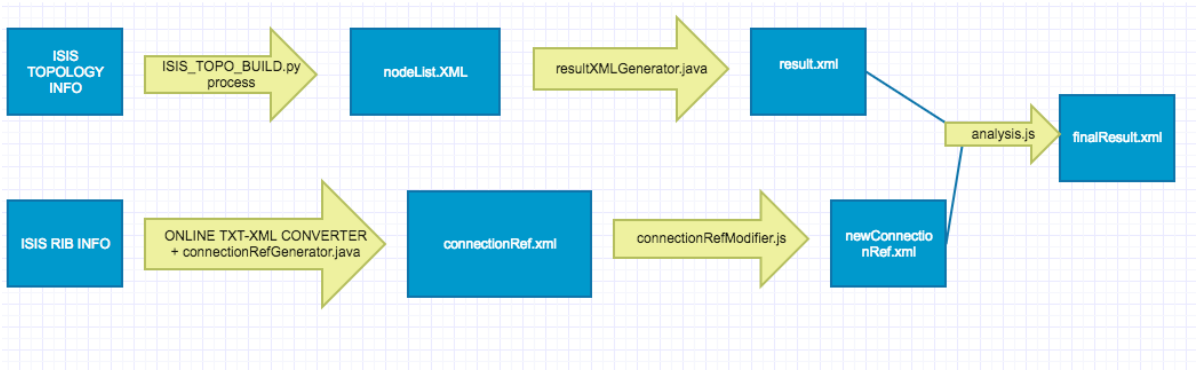
2. Output of command “show isis topology”

```
IS-IS paths to level-2 routers
System Id            Metric     Next-Hop             Interface   SNPA
NYCMng               60         DNVRng               Fa1/0       cc02.1cc0.0000
ATLAng               40         DNVRng               Fa1/0       cc02.1cc0.0000
ATLA-M5              50         DNVRng               Fa1/0       cc02.1cc0.0000
STTLng               --
SNVAng               10         SNVAng               Fa0/0       cc03.0a14.0000
DNVRng               10         DNVRng               Fa1/0       cc02.1cc0.0000
LOSAng               40         DNVRng               Fa1/0       cc02.1cc0.0000
KSCYng               20         DNVRng               Fa1/0       cc02.1cc0.0000
HSTNng               30         DNVRng               Fa1/0       cc02.1cc0.0000
CHINng               40         DNVRng               Fa1/0       cc02.1cc0.0000
IPLSng               30         DNVRng               Fa1/0       cc02.1cc0.0000
WASHng               50         DNVRng               Fa1/0       cc02.1cc0.0000
```

3. Output of command “show isis rib”

```
192.168.1.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
192.168.2.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.3.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
192.168.4.0/24
  [115/L2/20] via 192.168.1.2(FastEthernet1/0), from 3.3.3.3, tag 0, LSP[8/120]
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
192.168.5.0/24
  [115/L2/20] via 192.168.2.2(FastEthernet0/0), from 2.2.2.2, tag 0, LSP[4/121]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.6.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 4.4.4.4, tag 0, LSP[5/113]
192.168.7.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
192.168.8.0/24
  [115/L2/30] via 192.168.1.2(FastEthernet1/0), from 6.6.6.6, tag 0, LSP[3/119]
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
192.168.9.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 5.5.5.5, tag 0, LSP[11/105]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
192.168.10.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
192.168.11.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 15.15.15.15, tag 0, LSP[16/94]
192.168.12.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 9.9.9.9, tag 0, LSP[20/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 12.12.12.12, tag 0, LSP[24/93]
192.168.13.0/24
  [115/L2/40] via 192.168.1.2(FastEthernet1/0), from 8.8.8.8, tag 0, LSP[13/100]
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 7.7.7.7, tag 0, LSP[15/98]
192.168.14.0/24
  [115/L2/50] via 192.168.1.2(FastEthernet1/0), from 7.7.7.7, tag 0, LSP[15/98]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 10.10.10.10, tag 0, LSP[21/93]
192.168.15.0/24
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 15.15.15.15, tag 0, LSP[16/94]
  [115/L2/60] via 192.168.1.2(FastEthernet1/0), from 10.10.10.10, tag 0, LSP[21/93]
```

4. Data parsing procedure diagram



5. Processed file stage 1 for topology command(nodeList.xml)

```
<NodeList>
  <Node>
    <nodeId>NYCMng</nodeId>
    <Metric>60</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>ATLA-M5</nodeId>
    <Metric>50</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>SNVAng</nodeId>
    <Metric>10</Metric>
    <NextHop>SNVAng</NextHop>
  </Node>
  <Node>
    <nodeId>DNVRng</nodeId>
    <Metric>10</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>LOSAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>KSCYng</nodeId>
    <Metric>20</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>HSTNng</nodeId>
    <Metric>30</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>CHINng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>IPLSng</nodeId>
    <Metric>30</Metric>
    <NextHop>DNVRng</NextHop>
```

```
  </Node>
  <Node>
    <nodeId>WASHng</nodeId>
    <Metric>50</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
</NodeList>
```

6. Processed file stage 2 for topology command(result.xml)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<NodeList>
  <Node>
    <nodeId>NYCMng</nodeId>
    <Metric>60</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>ATLA-M5</fromId>
    <fromId>WASHng</fromId>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>HSTNng</fromId>
    <fromId>IPLSng</fromId>
  </Node>
  <Node>
    <nodeId>ATLA-M5</nodeId>
    <Metric>50</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>ATLAng</fromId>
    <fromId>LOSAng</fromId>
    <fromId>CHINng</fromId>
  </Node>
  <Node>
    <nodeId>SNVAng</nodeId>
    <Metric>10</Metric>
    <NextHop>SNVAng</NextHop>
  <fromId>STTLng</fromId>
  </Node>
  <Node>
    <nodeId>DNVRng</nodeId>
    <Metric>10</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>STTLng</fromId>
  </Node>
  <Node>
    <nodeId>LOSAng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>HSTNng</fromId>
    <fromId>IPLSng</fromId>
  </Node>
  <Node>
    <nodeId>KSCYng</nodeId>
    <Metric>20</Metric>
    <NextHop>DNVRng</NextHop>
  </Node>
  <Node>
    <nodeId>HSTNng</nodeId>
    <Metric>30</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>KSCYng</fromId>
  </Node>
  <Node>
    <nodeId>CHINng</nodeId>
    <Metric>40</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>HSTNng</fromId>
    <fromId>IPLSng</fromId>
  </Node>
  <Node>
```

```
    <nodeId>IPLSng</nodeId>
    <Metric>30</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>KSCYng</fromId>
  </Node>
  <Node>
    <nodeId>WASHng</nodeId>
    <Metric>50</Metric>
    <NextHop>DNVRng</NextHop>
  <fromId>ATLAng</fromId>
    <fromId>LOSAng</fromId>
    <fromId>CHINng</fromId>
  </Node>
</NodeList>
```

7. Processed file stage 1 for RIB command(connectionRef.xml)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<IpList>
  <IpRegion id="192.168.1.0/24">
    <fromId>3.3.3.3</fromId>
  </IpRegion>
  <IpRegion id="192.168.2.0/24">
    <fromId>2.2.2.2</fromId>
  </IpRegion>
  <IpRegion id="192.168.3.0/24">
    <fromId>DNVRng</fromId>
    <fromId>SNVAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.4.0/24">
    <fromId>DNVRng</fromId>
    <fromId>KSCYng</fromId>
  </IpRegion>
  <IpRegion id="192.168.5.0/24">
    <fromId>SNVAng</fromId>
    <fromId>LOSAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.6.0/24">
    <fromId>HSTNng</fromId>
    <fromId>LOSAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.7.0/24">
    <fromId>KSCYng</fromId>
    <fromId>HSTNng</fromId>
  </IpRegion>
  <IpRegion id="192.168.8.0/24">
    <fromId>KSCYng</fromId>
    <fromId>IPLSng</fromId>
  </IpRegion>
  <IpRegion id="192.168.9.0/24">
    <fromId>HSTNng</fromId>
    <fromId>ATLAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.10.0/24">
    <fromId>IPLSng</fromId>
    <fromId>ATLAng</fromId>
  </IpRegion>
  <IpRegion id="192.168.11.0/24">
    <fromId>ATLAng</fromId>
    <fromId>WASHng</fromId>
  </IpRegion>
  <IpRegion id="192.168.12.0/24">
         <fromId>ATLAng</fromId>
    <fromId>ATLA-M5</fromId>
  </IpRegion>
  <IpRegion id="192.168.13.0/24">
         <fromId>IPLSng</fromId>
    <fromId>CHINng</fromId>
  </IpRegion>
  <IpRegion id="192.168.14.0/24">
         <fromId>CHINng</fromId>
```

```
    <fromId>NYCMng</fromId>
  </IpRegion>
  <IpRegion id="192.168.15.0/24">
    <fromId>WASHng</fromId>
    <fromId>NYCMng</fromId>
  </IpRegion>
</IpList>
```

8. Processed file stage 2 for RIB command(newConnectionRef.xml)

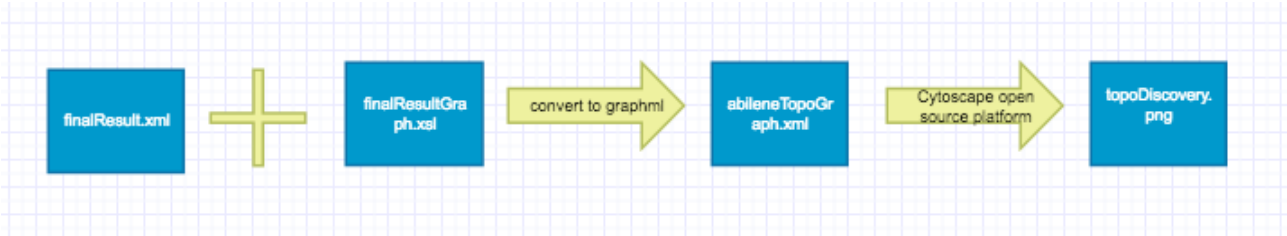
```
<NodeList>
  <Node>
    <nodeId>DNVRng</nodeId>
    <fromId>SNVAng</fromId>
  </Node>
  <Node>
    <nodeId>DNVRng</nodeId>
    <fromId>KSCYng</fromId>
  </Node>
  <Node>
    <nodeId>SNVAng</nodeId>
    <fromId>LOSAng</fromId>
  </Node>
  <Node>
    <nodeId>HSTNng</nodeId>
    <fromId>LOSAng</fromId>
  </Node>
  <Node>
    <nodeId>KSCYng</nodeId>
    <fromId>HSTNng</fromId>
  </Node>
  <Node>
    <nodeId>KSCYng</nodeId>
    <fromId>IPLSng</fromId>
  </Node>
  <Node>
    <nodeId>HSTNng</nodeId>
    <fromId>ATLAng</fromId>
  </Node>
  <Node>
    <nodeId>IPLSng</nodeId>
    <fromId>ATLAng</fromId>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <fromId>WASHng</fromId>
  </Node>
  <Node>
    <nodeId>ATLAng</nodeId>
    <fromId>ATLA-M5</fromId>
  </Node>
  <Node>
    <nodeId>IPLSng</nodeId>
    <fromId>CHINng</fromId>
  </Node>
  <Node>
    <nodeId>CHINng</nodeId>
    <fromId>NYCMng</fromId>
  </Node>
  <Node>
    <nodeId>WASHng</nodeId>
    <fromId>NYCMng</fromId>
  </Node>
</NodeList>
```

9. Combined command xml file(finalResult.xml)

```
<NodeList>
  <Node id="NYCMng">
```

```
    <fromId check="YES">WASHng</fromId>
    <fromId check="YES">CHINng</fromId>
  </Node>
  <Node id="ATLAng">
    <fromId check="YES">HSTNng</fromId>
    <fromId check="YES">IPLSng</fromId>
    <fromId check="YES">WASHng</fromId>
    <fromId check="YES">ATLA-M5</fromId>
  </Node>
  <Node id="ATLA-M5">
    <fromId check="YES">ATLAng</fromId>
  </Node>
  <Node id="SNVAng">
    <fromId>STTLng</fromId>
    <fromId check="YES">DNVRng</fromId>
    <fromId check="YES">LOSAng</fromId>
  </Node>
  <Node id="DNVRng">
    <fromId>STTLng</fromId>
    <fromId check="YES">SNVAng</fromId>
    <fromId check="YES">KSCYng</fromId>
  </Node>
  <Node id="LOSAng">
    <fromId check="YES">HSTNng</fromId>
    <fromId check="YES">SNVAng</fromId>
  </Node>
  <Node id="KSCYng">
    <fromId check="YES">DNVRng</fromId>
    <fromId check="YES">HSTNng</fromId>
    <fromId check="YES">IPLSng</fromId>
  </Node>
  <Node id="HSTNng">
    <fromId check="YES">KSCYng</fromId>
    <fromId check="YES">LOSAng</fromId>
    <fromId check="YES">ATLAng</fromId>
  </Node>
  <Node id="CHINng">
    <fromId check="YES">IPLSng</fromId>
    <fromId check="YES">NYCMng</fromId>
  </Node>
  <Node id="IPLSng">
    <fromId check="YES">KSCYng</fromId>
    <fromId check="YES">ATLAng</fromId>
    <fromId check="YES">CHINng</fromId>
  </Node>
  <Node id="WASHng">
    <fromId check="YES">ATLAng</fromId>
    <fromId check="YES">NYCMng</fromId>
  </Node>
  <Node id="STTLng">
    <fromId>SNVAng</fromId>
    <fromId>DNVRng</fromId>
  </Node>
</NodeList>
```

10. Topology visualization procedures



11. Graphxml pre-defined xsl file(finalResultGraph.xsl)

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

          <xsl:template match="NodeList">
                    <!--<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">-->
                              <key id="d0" for="edge" attr.name="weight" attr.type="double"/>
                              <graph id="nodeConnection" edgedefault="undirected">
                                        <xsl:for-each select="Node">
                                                  <xsl:variable name="node" select="." />
                                                  <node id="{$node/@id}"/>
                                        </xsl:for-each>
                                        <xsl:for-each select="Node">
                                                  <xsl:variable name="dest" select="." />
                                                  <xsl:for-each select="fromId">
                                                            <xsl:variable name="src" select="."/>
                                                            <edge source="{$dest/@id}" target = "{$src}">
                                                                      <data key="d0">
                                                                                <xsl:value-of select="10.0"/>
                                                                      </data>
                                                            </edge>
                                                  </xsl:for-each>
                                        </xsl:for-each>
                              </graph>
                    <!--</graphml>-->
          </xsl:template>
</xsl:stylesheet>
```

12. Final verision of graphxml file(abileneTopology.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
<key attr.type="double" attr.name="weight" for="edge" id="d0"/>
<graph edgedefault="undirected" id="nodeConnection">
          <node id="NYCMng"/>
          <node id="ATLAng"/>
          <node id="ATLA-M5"/>
          <node id="SNVAng"/>
          <node id="DNVRng"/>
          <node id="LOSAng"/>
          <node id="KSCYng"/>
          <node id="HSTNng"/>
          <node id="CHINng"/>
          <node id="IPLSng"/>
          <node id="WASHng"/>
          <node id="STTLng"/>
          <edge target="WASHng" source="NYCMng">
                    <data key="d0">10</data>
          </edge>
          <edge target="CHINng" source="NYCMng">
                    <data key="d0">10</data>
          </edge>
          <edge target="HSTNng" source="ATLAng">
                    <data key="d0">10</data>
          </edge>
          <edge target="IPLSng" source="ATLAng">
                    <data key="d0">10</data>
          </edge>
          <edge target="WASHng" source="ATLAng">
                    <data key="d0">10</data>
          </edge>
          <edge target="ATLA-M5" source="ATLAng">
                    <data key="d0">10</data>
          </edge>
          <edge target="ATLAng" source="ATLA-M5">
                    <data key="d0">10</data>
          </edge>
          <edge target="STTLng" source="SNVAng">
                    <data key="d0">10</data>
```

```
        </edge>
        <edge target="DNVRng" source="SNVAng">
                <data key="d0">10</data>
        </edge>
        <edge target="LOSAng" source="SNVAng">
                <data key="d0">10</data>
        </edge>
        <edge target="STTLng" source="DNVRng">
                <data key="d0">10</data>
        </edge>
        <edge target="SNVAng" source="DNVRng">
                <data key="d0">10</data>
        </edge>
        <edge target="KSCYng" source="DNVRng">
                <data key="d0">10</data>
        </edge>
        <edge target="HSTNng" source="LOSAng">
                <data key="d0">10</data>
        </edge>
        <edge target="SNVAng" source="LOSAng">
                <data key="d0">10</data>
        </edge>
        <edge target="DNVRng" source="KSCYng">
                <data key="d0">10</data>
        </edge>
        <edge target="HSTNng" source="KSCYng">
                <data key="d0">10</data>
        </edge>
        <edge target="IPLSng" source="KSCYng">
                <data key="d0">10</data>
        </edge>
        <edge target="KSCYng" source="HSTNng">
                <data key="d0">10</data>
        </edge>
        <edge target="LOSAng" source="HSTNng">
                <data key="d0">10</data>
        </edge>
        <edge target="ATLAng" source="HSTNng">
                <data key="d0">10</data>
        </edge>
        <edge target="IPLSng" source="CHINng">
                <data key="d0">10</data>
        </edge>
        <edge target="NYCMng" source="CHINng">
                <data key="d0">10</data>
        </edge>
        <edge target="KSCYng" source="IPLSng">
                <data key="d0">10</data>
        </edge>
        <edge target="ATLAng" source="IPLSng">
                <data key="d0">10</data>
        </edge>
        <edge target="CHINng" source="IPLSng">
                <data key="d0">10</data>
        </edge>
        <edge target="ATLAng" source="WASHng">
                <data key="d0">10</data>
        </edge>
        <edge target="NYCMng" source="WASHng">
                <data key="d0">10</data>
        </edge>
        <edge target="SNVAng" source="STTLng">
                <data key="d0">10</data>
        </edge>
        <edge target="DNVRng" source="STTLng">
                <data key="d0">10</data>
        </edge>
</graph>
</graphml>
```

13. Final topology diagram