

2017

ISEConf: A management tool for the ISEAGE Environment to allow users to create a custom mock internet for testbed research

Jeffrey Lincoln Neel
Iowa State University

Follow this and additional works at: <https://lib.dr.iastate.edu/etd>



Part of the [Computer Engineering Commons](#)

Recommended Citation

Neel, Jeffrey Lincoln, "ISEConf: A management tool for the ISEAGE Environment to allow users to create a custom mock internet for testbed research" (2017). *Graduate Theses and Dissertations*. 15589.
<https://lib.dr.iastate.edu/etd/15589>

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

ISEConf: A management tool for the ISEAGE Environment to allow users to create a custom mock internet for testbed research

by

Jeffrey Lincoln Neel

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co-majors: Information Assurance; Computer Engineering

Program of Study Committee:
Doug Jacobson, Major Professor
James Davis
George Amariuca

Iowa State University

Ames, Iowa

2017

Copyright © Jeffrey Lincoln Neel, 2017. All rights reserved.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	iv
NOMENCLATURE	v
ACKNOWLEDGMENTS	vi
ABSTRACT.....	vii
CHAPTER I INTRODUCTION: THE ISERINK ENVIRONMENT	1
Overview of ISERink.....	1
Current Problems with ISERink	6
CHAPTER II RELATED WORKS	7
DETER	7
Other Solutions	8
CHAPTER III PROPOSED FEATURES TO ISERINK.....	9
Automatic ISEFlow Config Generator	11
ISEFlow Visualization	11
ISEFlow Statistics.....	11
CHAPTER IV FEATURE IMPLIMENTATIONS.....	13
ISEFlow Config File Background	13
Automatic ISEFlow Config Generator	23
ISEFlow Visualization	25
ISEFlow Statistics.....	27
CHAPTER V FUTURE WORKS.....	30
Improved Visualizations	30
Adding New Deices to ISEFlow.....	30
Federation	31
CHAPTER VI SUMMARY AND CONCLUSIONS	32
REFERENCES	33

APPENDIX A OLD FLOW CONFIG	34
APPENDIX B NEW FLOW CONFIG	44

LIST OF FIGURES

	Page
Figure 1 Overview of the ISERink Environment.....	3
Figure 2 New ISERink Architecture	10
Figure 3 Architecture of a ISEFlow Board	15
Figure 4 Global Definition Example.....	16
Figure 5 Board Definition Connection Example.....	17
Figure 6 Device Definition for outside device	18
Figure 7 Device Definition for inside and internal device	20
Figure 8 Outside Device Definition for Tap Board.....	22
Figure 9 Database relations	23
Figure 10 Descriptions of database tables.....	25
Figure 11 Creating a new Device	26
Figure 12 Visualization of the Configured Environment.....	28

NOMENCLATURE

ISEAGE	Internet-Scale Event and Attack Generation Environment
NIC	Network Interface Connection
VM	Virtual Machine
OS	Operating System
Host	A Physical Server
Hypervisor	OS that allows users to create VM's that directly use the host resources
ESXi	VMware's Hypervisor
WAN	Wide Area Network
LAN	Local Area Network

ACKNOWLEDGMENTS

I would like to thank my committee chair, Dr. Doug Jacobson, and my committee members, Dr. Jim Davis and Dr. George Amariuca, for their support during my research. I would also like to thank Dr. Jacobson for allowing me to work with the ISEAGE research lab as an undergraduate. The work I did there was one of the most memorable parts of my undergraduate experience. In addition, I would also like to thank my family who has supported me throughout my college career and without whom, this thesis would not have been possible.

ABSTRACT

The ISEAGE (Internet-Scale Event and Attack Generation Environment) has grown and changed since it was originally created. ISEAGE was the original testbed environment that was developed on physical hardware. Since virtualization has become more widely available ISEAGE has shifted into a distributable environment that anyone working in the academic field can download and use for free, this environment is known as ISERink. This paper details the improvements that I have made while working with Dr. Jacobson to add to the core of the ISERink environment. The improvements are intended to help the ISERink environment be more customizable by upgrading the config file format to a more human readable format, creating an application to automatically create config files that could be easily expandable in the future, and adding features such as visualization of the networks to help the end user of the environment. These features are planned to be released with ISERink v2 in the fall 2017.

CHAPTER I

INTRODUCTION: THE ISERINK ENVIRONMENT

Overview of ISERink

ISERink is a cyber security playground that is available to educational institutions to help teach students about cyber security. ISERink provides a network environment that mimics the internet. This allows students to setup mock networks and safely attack other machines in an isolated environment. The ISERink environment consists of a set of 13 virtual machines that allow it to create this mock internet. These virtual machines consist of four main groups based on their function: Keyhole, Testing, Support and ISEFlow. Keyhole machines allow users to access the real internet via HTTP, HTTPS, and FTP. Testing machines are used to make sure the environment is setup correctly, while Supporting machines contain additional virtual machines that can support additional features for the environment. Lastly, ISEFlow machines are the machines that run ISEFlow, a program written by Dr. Jacobson at Iowa State University that handles all routing in the mock internet, or are essential for its execution. ISEFlow is configurable based off a config file this allows the user to configure the mock internet with custom routing of packets.

At Iowa State University, the ISEAGE research lab has been hosting Cyber Defense Competitions on ISERink for over a decade. This is where the nomenclature for the blue and red-green boards comes from. In the typical setup board one and two are blue boards, which would be for the blue teams acting as cyber security professionals at a company. Board three is a red-green board and is for red and green team. Red team are the attackers and green team are the end

users. These teams typically share a board so they can share IP ranges, this prevents blue teams from blocking users based on the origin of the request.

A diagram of a standard ISERink deployment on VMware ESXi (this can be deployed on any hypervisor) is detailed in Figure 1 below. In this figure, red represents Virtual machines, yellow represents internal vSwitches, and green represents vSwitches that are mapped to physical NIC's on the host.

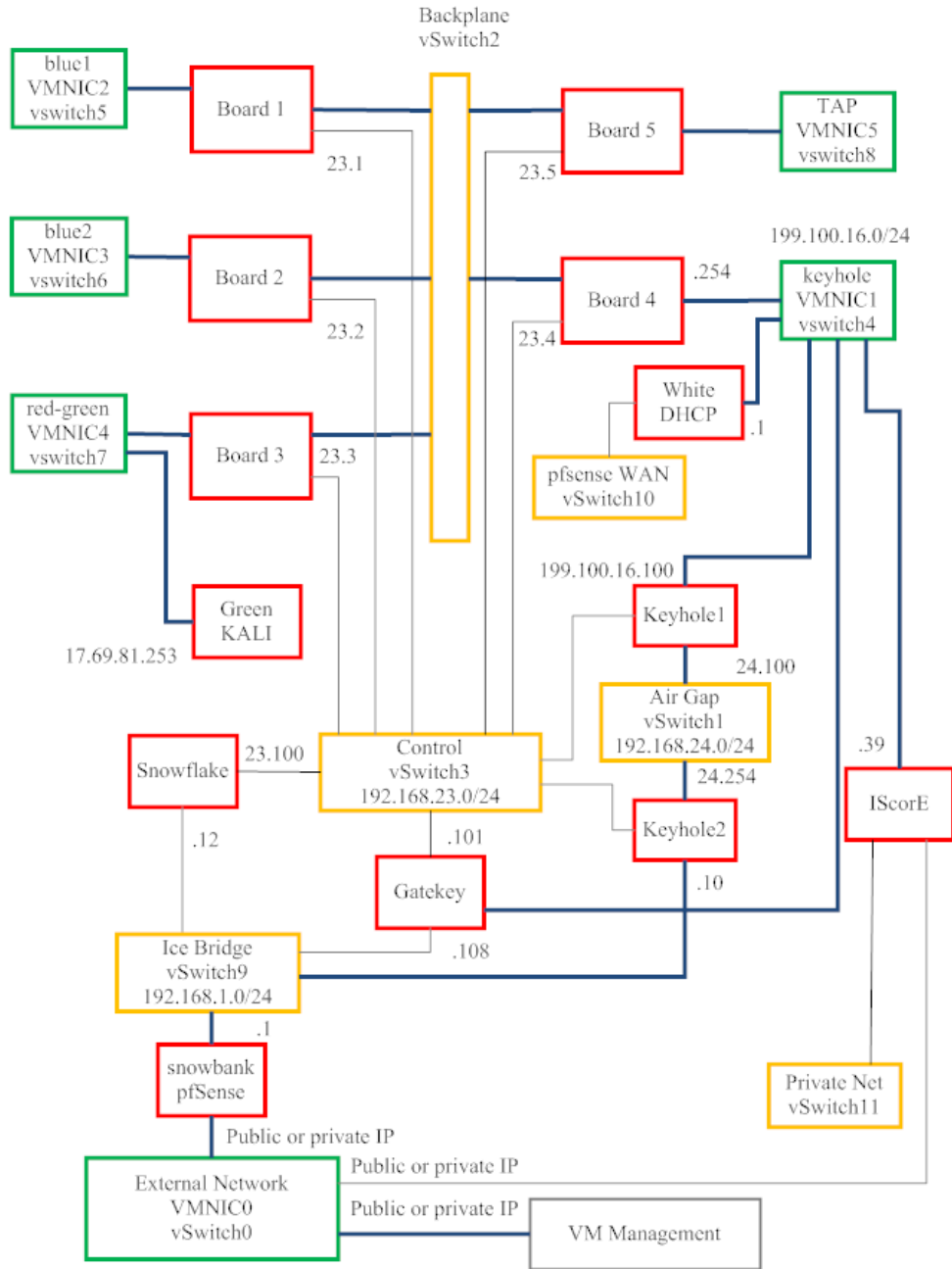


Figure 1: Overview of the ISERink Environment

Keyhole machines

The Keyhole machines are two squid proxy servers that proxy all requests that are made and bound for the “real internet.” The allowed protocols consist of HTTP, HTTPS, and FTP.

These protocols let users inside the environment use a web browser to access the internet and use a package manager to update the systems inside of the network. While this system is not required for the mock internet to function it helps support the cyber security playground and allows users to preform research from the internal VM's and pull down the latest patches for their systems.

Testing machines

The Testing machines consist of Green Kali and Gatekey systems. The Green Kali and Gatekey virtual machines are used purely for testing and configuring systems. The Green Kali is a Kali Linux machine inside of the cyber security playground that is attached to one of the red-green networks on board three. This board is a piece of ISEFlow that we will touch on later. Gatekey is a Linux Mint VM that is used for configuring pfSense on the LAN of Snowbank. This is primarily needed to set the IP addresses of Snowbank to configure the WAN of the environment. After the playground has been setup and tested these VM's can typically be powered off to conserve resources on the host.

Support machines

As mentioned previously, ISERink is used to host Cyber Defense Competitions at Iowa State University. Over the past decade Iowa State has developed a scoring system that is used for the competition called IScoreE. The IScoreE Support machine is an optional support system that aids in hosting Cyber Defense Competitions. A few of its features include team management, automatic scamming/grading of services, and documentation management just to name a few. Full details on IScoreE can be found on the official documentation page at <https://docs.iseage.org/iscore/admin/latest/>.

ISEFlow machines

Finally, we get to the main portion of ISERink, ISEFlow. As previously mentioned ISEFlow is a program written by Dr. Doug Jacobson. This program handles all the routing of packets within the environment and allows it to mimic the public internet by using real world IPv4 addresses and simulating hoops in the environment through software defined networking devices. ISEFlow is a scalable solution that allows the modeling of many networks. The system from Figure 1 is a five board ISEFlow system, but the number of boards in an ISERink can be scaled up to as many as needed. Each board handles one to twenty IP ranges. For example, in terms of Cyber Defense Competitions one of these IP ranges on the blue boards maps to one blue team. All boards are running the ISEFlow program and handling the routing.

There are three types of boards that exist in ISEFlow. These boards perform different tasks that are important to how the overall architecture of ISERink works. The blue and red-green boards are internal boards that allow machines to connect to the environment. The second type of board is the Proxy board, this is what allows users to connect to the real internet through Keyhole. The Proxy board takes all traffic bound for the real internet and sends it off to Keyhole machines to make the request. The third type of board is Tap. The Tap board allows packet captures for all traffic that is being transferred over ISEFlow. This is a great tool for researchers since it allows them to monitor the networks and capture any data they may need to reproduce an experiment.

All boards work with Snowflake. Snowflake is the virtual machine that manages ISEFlow and hosts a network file share that all the boards mount and use for their configurations. These configurations are binary compiled files that ISEFlow reads. Snowflake provides the

ability to compile config files using `mk_map` for the old config file format or `mk_flow` for the new config file format. There is a web application that is hosted on Snowflake that is the current control center for ISEFlow.

The Current Problems in ISERink

Looking at ISEFlow itself there is a large amount of potential. The ability to configure a mock internet as a cyber security playground has a tremendous amount of uses, such as training exercises, modeling networks, and research. Unfortunately, when people look into ISEFlow it becomes a major challenge to understand and write ISEFlow config files in the old format (Appendix A). With this in mind the new config file format was proposed. The new config file format works with the idea of making the config files more easily readable for humans. However, as seen in Appendix B the new config file format can still be overwhelming to look at. The current environment also lacks any form of visualization as to what the networks look like within the environment. If a user wanted to know what the entire environment looked like they would have two options: run a traceroute to every IP range's gateway or read through the config files, either way the user would have to map this by hand to get a complete visualization of the network.

CHAPTER II

RELATED WORKS

DETER

ISERink is not the only cyber security testbed environment. DETER is another testbed technology that is hosted at the University of Southern California's Information Sciences Institute at the University of California at Berkley. This project came from an NSF and DHS grant in the early 2000's. The DETER environment completely isolates communications to the real internet besides a SSH connection that is used to manage the experiment. The DETER project also has many features built in for users to use, such as traffic generation and modeling.

DETER allows anyone to apply for access to use their environment free of charge for research and classes. This allows for a variety of users. It seems that the main uses for the environment are education and research, such as modeling attack traffic and performing predictive modeling on results.

DETER is built upon the Emulab codebase. This was extended to allow more complex simulations at a larger scale. The modifications also allowed more specific configurations that apply to cyber security. This allows for a lot of interesting features that Emulab has. One of these features is allowing federated environments. This was made possible due to Emulab and now the DETER environment can federate with any environment running the Emulab software. while DETER took advantage of tools like Emulab, ISERinks system was developed by Iowa State University at Iowa State.

Other Solutions

As opposed to using technologies like DETER and ISERink that are developed testbed environments the other solution is creating a network of physical routers. This would require buying a large set of routers and physically connecting them together. While this may be the most straight-forward way to get an environment set up it is a costly approach. This requires a physical location to hook up all the routers and machines, while DETER could be accessed remotely and an ISERink could be hosted on a single server.

CHAPTER III

PROPOSED FEATURES TO ISERINK

This project spawned from looking at the complexity of customizing an ISERink environment and thinking of ways to make this simpler for anyone that wants to use the ISERink environment. The conclusion made was that the Snowflake web application does not meet the current needs of the environment since it was distributed outside of Iowa State University. As there are now ISERinks being hosted across the US and even in other countries, it is desirable that users are able to customize the environment to meet a wide number of needs. An application called ISEConf was then created. ISEConf will be the new central location for users to manage and configure ISEFlow from a simple to use web interface that will be hosted on a new virtual machine connected to the control network as seen in Figure 2. This application is written in Django, which is a well-supported python web framework that is also used for IScoreE. This was chosen to create consistency across the environment and allow for the ISEAGE lab to update ISEConf as needed. The following sections provide more details on the features of ISEConf.

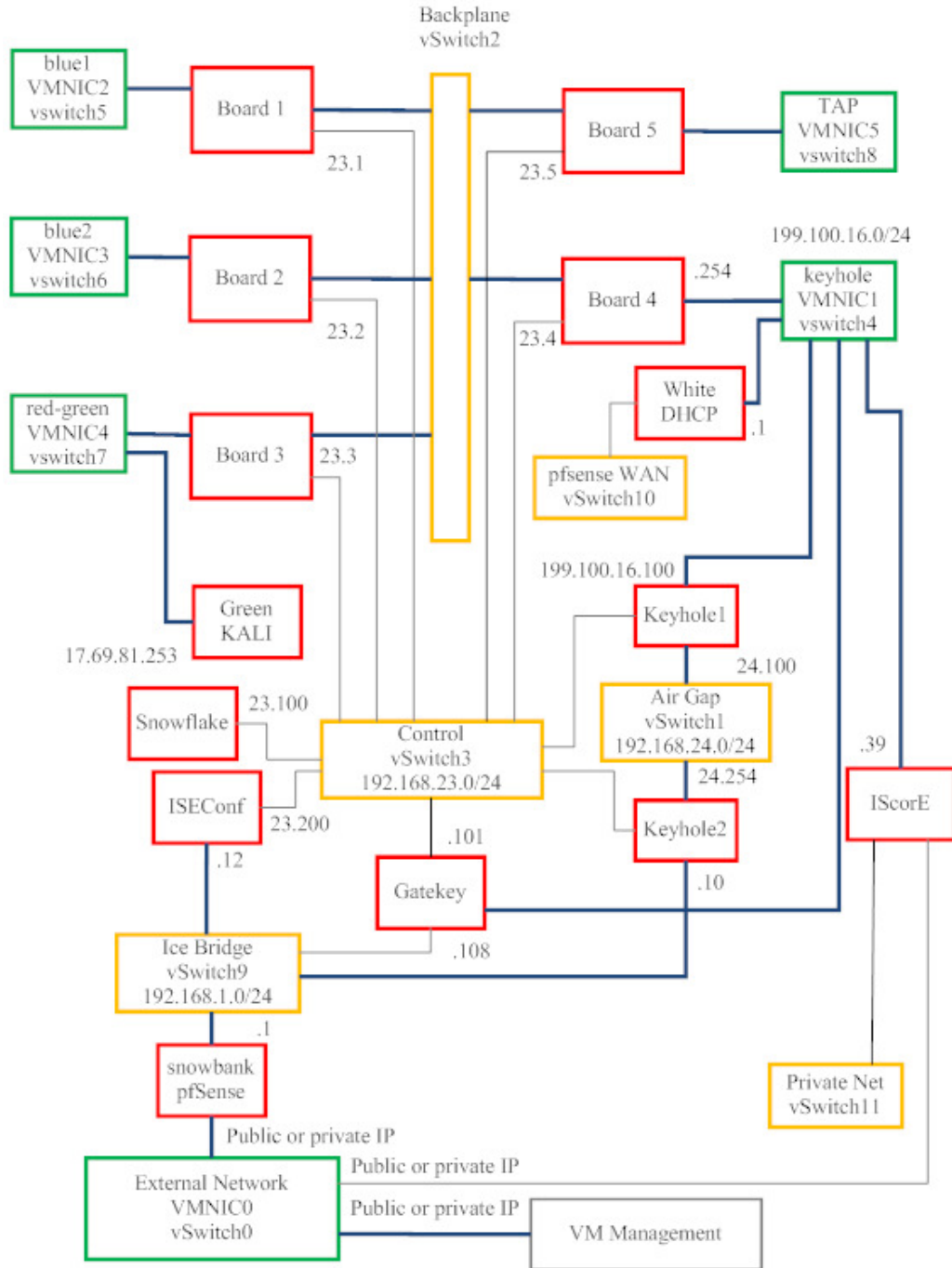


Figure 2: New ISERink Architecture

Automatic ISEFlow Config Generator

The main goal of ISEConf is to abstract away the complexity of the config files by using an automated process. This is done by first having the user define the connections, routing, and virtual routers to be used in ISEConf, followed by selecting which boards the routers and networks will be on. After the user has defined what the network will look like all of the details are stored in a database. The user can easily create a config file from the database entries and upload this directly to Snowflake, where it will be compiled and pushed out to the boards. This is all done in the background, so the end user only needs to perform a few simple actions after defining the network to have a working ISEFlow with the new network.

ISEFlow Visualization

The old ISERink setup had no way of showing what the network looked like without looking at the config files. ISEConf will add the ability to easily visualize ISEFlow. This adds the ability to see all the virtual routers that are defined in the environment and the IP addresses of each hop in the network. This type of visualization is essential for research using the ISERink environment and is useful in Cyber Defense Competitions for showing how traffic is being routed between teams. This will be a very useful tool to have for the environment and this is a tool that could potentially be expanded on to create a lot more detailed visualizations that could include real-time packet flows.

ISEFlow Statistics

The ISEFlow process running on each board keeps track of statistics about ISEFlow and how many packets are running through each of the virtual routers. The old implementation had a

simple table that showed the packets in and out of each IP range in the ISEFlow system. ISEConf will pull all statistics and show the change in the packet flow over time. The visualization of the packet change will help researchers see how data is flowing in the environment over time.

CHAPTER IV

FEATURE IMPLIMENTATION

As previously stated, ISEConf is a Django python web application that is hosted on Debian Stretch. This project is attempting to mimic as closely as possible the ISERink production systems, languages, frameworks, and coding standards. This allows the application to be easily updated and modified to meet future needs of ISERink. ISEConf was designed with future needs in mind by utilizing modular and separated components that can be easily removed or added as needed.

ISEFlow Configuration file background

Before getting too far into the technical view of how these features were implemented a more technical view of ISEFlow is required. The following sections give a very in depth description of how the internals of ISEFlow and the config files work.

Snowflake

Although Snowflake is the central point for managing config files it is still the simplest portion of the ISEFlow system. Snowflake is not running any of the ISEFlow software, but it does have the capability to compile the config files. This program `mk_flow` will compile the new config file format to binary files that ISEFlow uses to run each board. All of these files are shared though a network file share that consists of the Scrat users home directory. This directory

contains all binary and configurations that the boards need to load the config files and run ISEFlow.

This makes it fairly easy to spin up new boards in an environment since it only needs to be able to mount the network file share and have few config files in place that tell it what board it is and meets the dependencies for ISEFlow.

ISEFlow boards internals

When examining the internal workings of the boards one can see how the config files could get complex. ISEFlow boards use software defined networking devices to direct traffic in the environment. There are four types of networking devices that the new config file format allows in ISEFlow. These devices are Router, Cloud, Link, and Network. Routers are the only type of device that is currently supported in V1.0 of the new config file format for ISEFlow, so this is the main device that will be dealt with and tested.

A board can contain a series of devices. In the typical configuration, the blue boards contain 3 hops with multiple IP ranges on each board. For each of the devices on the board interfaces and routes need to be created to designate how to route the traffic within the board by connecting a series of interfaces together. Figure 3 below details the internals of a board to give a visual for the connections within the device.

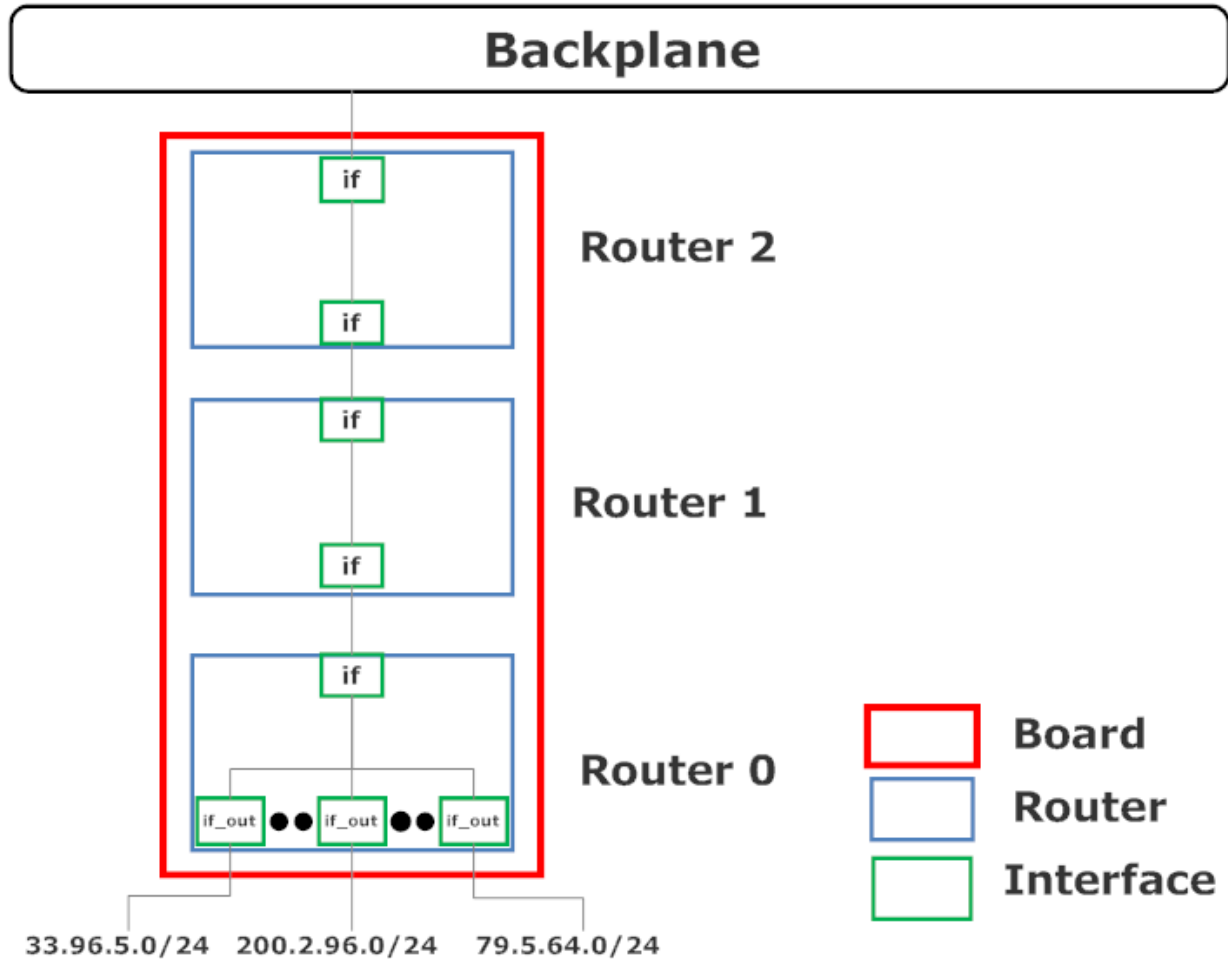


Figure 3: Internal Architecture of a ISEFlow Board

The diagram shows how the connections between each device is mapped. Looking at this board we can see that it has three devices that are routers. These routers contain sets of interconnected interfaces. You can think of these connections as physical network cables. There are two types of interfaces that ISEFlow understands, interface and interface_out. The interface type is for connecting devices to each other or to the backplane. Interface_out is for mapping to the IP range that is used for the mock internet. In this case the three IP ranges, 33.96.5.0, 200.2.96.0 and 79.5.64.0, would all be IP ranges that either physical or virtual machines could connect to and be placed inside of the cyber security playground.

New config file format

As seen from comparing the config file formats, they are fairly similar in terms of length and syntax, but the new config file format tends to be easier to read after understanding the basics of ISEFlow. There are two main sections of the new config file format: the global definitions and the board definitions. Figure 4 below is an example of the global definitions.

```

VER=1,0
##### Global Definitions #####
Globals= {
  BPnet=0,130.175.1.0,24
  board=1,3,1,130.175.1.1
  board=2,3,1,130.175.1.2
  board=3,3,1,130.175.1.3
  board=4,1,1,130.175.1.4
  board=5,1,1,130.175.1.5
  DLINK=IP
  Parms = {
    name=generic_link
  /parms

  GLINK=1,IP
  Parms = {
    name=IP link with loss
    # 1 percent loss
    loss=1.0
  /parms
/global

```

Figure 4: Global Definition Example

The global definitions define some basic principles that are used throughout the config file. The first is the version of the config file that is being used. Version 1.0 is the current released version. The global definitions define the backplane that all the boards are connected to as well as each of the boards. A board definition is what defines the board parameters used to customize each individual environment, such as route tables, connections, and devices. The board definition is in the form 'board=,board_number, number_of_devices, interface_on_backplane, IP_on_backplane'. The link's in the global section defines the global links that can be used

throughout the file. This can be used to do things like introduce packet loss to a network. Figure 5 starts to show the configuration that is needed for a single board.

```
##### Board 1 #####
board=1
connections={
  R0,0 => O
  R0,1 => O
  R0,2 => O
  R0,3 => O
  R0,4 => O
  R0,5 => O
  R0,6 => O
  R0,7 => O
  R0,8 => O
  R0,9 => O
  R0,10 => O
  R0,11 => O
  R0,12 => O
  R0,13 => O
  R0,14 => O
  R0,15 => O
  R0,16 => O
  R0,17 => O
  R0,18 => O
  R0,19 => O
  R0,20 => R1, 0, IP
  R1,1 => R2,0 , L
  R2,1 => B0, G, IP
}/connections
```

Figure 5: Board Definition Connection Example

Figure 5 shows the start of a board definition. The first thing in any board definition is ‘board= board_number’ this allows mk_flow to separate the different binaries for each board in the environment. Next are the connections, the connections tell how the devices are interconnected and what interface they are using to connect. In Figure 5, the left of the “=>” shows what router and interface the connection is being defined on and the right side shows where the connection is going to. This is either represented by a “O” which tells that this is an

interface_out to an IP range or it points to the next device or backplane, such as “R1, 0, IP.”.

After the connections are defined each device needs to be defined, Figure 6 shows the definition for the first device, router 0, on board 1.

```

device=router,0
  if_out=0,33.96.5.0,24,00:00:0c:01:00:00
  if_out=1,200.2.96.0,24,00:00:0c:01:00:01
  if_out=2,79.5.64.0,24,00:00:0c:01:00:02
  if_out=3,104.190.101.0,24,00:00:0c:01:00:03
  if_out=4,6.87.159.0,24,00:00:0c:01:00:04
  if_out=5,73.19.46.0,24,00:00:0c:01:00:05
  if_out=6,89.65.43.0,24,00:00:0c:01:00:06
  if_out=7,168.84.5.0,24,00:00:0c:01:00:07
  if_out=8,175.86.94.0,24,00:00:0c:01:00:08
  if_out=9,128.45.126.0,24,00:00:0c:01:00:09
  if_out=10,96.2.101.0,24,00:00:0c:01:00:10
  if_out=11,2.104.163.0,24,00:00:0c:01:00:11
  if_out=12,201.53.2.0,24,00:00:0c:01:00:12
  if_out=13,61.14.60.0,24,00:00:0c:01:00:13
  if_out=14,198.45.10.0,24,00:00:0c:01:00:14
  if_out=15,64.91.37.0,24,00:00:0c:01:00:15
  if_out=16,13.46.179.0,24,00:00:0c:01:00:16
  if_out=17,164.85.26.0,24,00:00:0c:01:00:17
  if_out=18,185.50.64.0,24,00:00:0c:01:00:18
  if_out=19,178.231.85.0,24,00:00:0c:01:00:19
  if=20,130.170.1.100,24
  ##### R Tables #####
  r_table=33.96.5.0,24,33.96.5.254,0
  r_table=200.2.96.0,24,200.2.96.254,1
  r_table=79.5.64.0,24,79.5.64.254,2
  r_table=104.190.101.0,24,104.190.101.254,3
  r_table=6.87.159.0,24,6.87.159.254,4
  r_table=73.19.46.0,24,73.19.46.254,5
  r_table=89.65.43.0,24,89.65.43.254,6
  r_table=168.84.5.0,24,168.84.5.254,7
  r_table=175.86.94.0,24,175.86.94.254,8
  r_table=128.45.126.0,24,128.45.126.254,9
  r_table=96.2.101.0,24,96.2.101.254,10
  r_table=2.104.163.0,24,2.104.163.254,11
  r_table=201.53.2.0,24,201.53.2.254,12
  r_table=61.14.60.0,24,61.14.60.254,13
  r_table=198.45.10.0,24,198.45.10.254,14
  r_table=64.91.37.0,24,64.91.37.254,15
  r_table=13.46.179.0,24,13.46.179.254,16
  r_table=164.85.26.0,24,164.85.26.254,17
  r_table=185.50.64.0,24,185.50.64.254,18
  r_table=178.231.85.0,24,178.231.85.254,19
# To Inside
  r_table=0.0.0.0,24,130.170.1.254,20
  Parms = {
    name= Board 1 Outside Router 0
  }
  /parms
/router

```

Figure 6: Device Definition for outside device

At first glance it looks like there is a lot going on in the above device definition, but this can simply be broken down to three sections: declaring interfaces, route tables to outside devices, and routes to internal devices that may pass traffic to another board. These interface definitions are very simple. It is simply declaring if this is an interface or an interface_out. Each of these interfaces requires an identification number, IP address, and a netmask. The interface_out also requires a MAC address that will be used for this interface. The route tables simply tells the board where the traffic needs to be directed to. The first set of route tables will send all data back to the IP range that it is being sent to. If the packet is trying to reach a IP range that is not specified in one of these route tables it will pass this traffic up to the next device until it gets passed to the next board where it will get routed to the destination. This is the same way that the traffic is handled in all the routers in a board. Figure 7 shows the device definitions for routers one and two.

```

device=router,1
  if=0,130.170.1.254,24
  if=1,131.10.1.100,24
  . . . . . #### R Tables ####
  r_table=33.96.5.0,24,131.10.1.100,0
  r_table=200.2.96.0,24,131.10.1.100,0
  r_table=79.5.64.0,24,131.10.1.100,0
  r_table=104.190.101.0,24,131.10.1.100,0
  r_table=6.87.159.0,24,131.10.1.100,0
  r_table=73.19.46.0,24,131.10.1.100,0
  r_table=89.65.43.0,24,131.10.1.100,0
  r_table=168.84.5.0,24,131.10.1.100,0
  r_table=175.86.94.0,24,131.10.1.100,0
  r_table=128.45.126.0,24,131.10.1.100,0
  r_table=96.2.101.0,24,131.10.1.100,0
  r_table=2.104.163.0,24,131.10.1.100,0
  r_table=201.53.2.0,24,131.10.1.100,0
  r_table=61.14.60.0,24,131.10.1.100,0
  r_table=198.45.10.0,24,131.10.1.100,0
  r_table=64.91.37.0,24,131.10.1.100,0
  r_table=13.46.179.0,24,131.10.1.100,0
  r_table=164.85.26.0,24,131.10.1.100,0
  r_table=185.50.64.0,24,131.10.1.100,0
  r_table=178.231.85.0,24,131.10.1.100,0
# To Inside
  r_table=0.0.0.0,24,131.10.1.254,1
  Parms = {
  . . . . . name= Board 1 Internal Router 1
  }
  /parms
/router
device=router,2
  if=0,131.10.1.254,24
  if=1,130.175.1.1,24
  . . . . . #### R Tables ####
  r_table=33.96.5.0,24,131.10.1.100,0
  r_table=200.2.96.0,24,131.10.1.100,0
  r_table=79.5.64.0,24,131.10.1.100,0
  r_table=104.190.101.0,24,131.10.1.100,0
  r_table=6.87.159.0,24,131.10.1.100,0
  r_table=73.19.46.0,24,131.10.1.100,0
  r_table=89.65.43.0,24,131.10.1.100,0
  r_table=168.84.5.0,24,131.10.1.100,0
  r_table=175.86.94.0,24,131.10.1.100,0
  r_table=128.45.126.0,24,131.10.1.100,0
  r_table=96.2.101.0,24,131.10.1.100,0
  r_table=2.104.163.0,24,131.10.1.100,0
  r_table=201.53.2.0,24,131.10.1.100,0
  r_table=61.14.60.0,24,131.10.1.100,0
  r_table=198.45.10.0,24,131.10.1.100,0
  r_table=64.91.37.0,24,131.10.1.100,0
  r_table=13.46.179.0,24,131.10.1.100,0
  r_table=164.85.26.0,24,131.10.1.100,0
  r_table=185.50.64.0,24,131.10.1.100,0
  r_table=178.231.85.0,24,131.10.1.100,0
#Route to Proxy Board
  r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
  r_table=0.0.0.0,16,130.175.1.5,1
  Parms = {
  . . . . . name= Board 1 Inside Router 2
  }
  /parms
/router

```

Figure 7: Device Definition for inside and internal device

As we can see in Figure 7 these device definitions can get complex with where traffic is going in the environment. Not only do the devices need to keep track of where they are sending the data, but the internal devices need to keep track of what board they will be sending packets to. As the number of boards increases so does the complexity of the config files. This is especially true for the Tap and Proxy boards. These boards need to keep track of every IP range and what boards they are on. Figure 8 shows the complexity that a single device can get to when there are 45+ IP ranges in the network.

```

device=router,0
  if_out=0,10.0.0.0,16,00:00:0c:05:00:00
  if=1,130.175.1.5,24
  ..... #### R Tables ####
  r_table=10.0.0.0,16,10.0.255.254,0
#Route to Internal Boards
  r_table=33.96.5.0,24,130.175.1.1,1
  r_table=200.2.96.0,24,130.175.1.1,1
  r_table=79.5.64.0,24,130.175.1.1,1
  r_table=104.190.101.0,24,130.175.1.1,1
  r_table=6.87.159.0,24,130.175.1.1,1
  r_table=73.19.46.0,24,130.175.1.1,1
  r_table=89.65.43.0,24,130.175.1.1,1
  r_table=168.84.5.0,24,130.175.1.1,1
  r_table=175.86.94.0,24,130.175.1.1,1
  r_table=128.45.126.0,24,130.175.1.1,1
  r_table=96.2.101.0,24,130.175.1.1,1
  r_table=2.104.163.0,24,130.175.1.1,1
  r_table=201.53.2.0,24,130.175.1.1,1
  r_table=61.14.60.0,24,130.175.1.1,1
  r_table=198.45.10.0,24,130.175.1.1,1
  r_table=64.91.37.0,24,130.175.1.1,1
  r_table=13.46.179.0,24,130.175.1.1,1
  r_table=164.85.26.0,24,130.175.1.1,1
  r_table=185.50.64.0,24,130.175.1.1,1
  r_table=178.231.85.0,24,130.175.1.1,1
  r_table=64.39.3.0,24,130.175.1.2,1
  r_table=201.203.200.0,24,130.175.1.2,1
  r_table=64.5.53.0,24,130.175.1.2,1
  r_table=49.49.33.0,24,130.175.1.2,1
  r_table=33.96.50.0,24,130.175.1.2,1
  r_table=76.5.61.0,24,130.175.1.2,1
  r_table=45.65.85.0,24,130.175.1.2,1
  r_table=159.75.35.0,24,130.175.1.2,1
  r_table=178.98.45.0,24,130.175.1.2,1
  r_table=197.45.10.0,24,130.175.1.2,1
  r_table=5.0.80.0,24,130.175.1.2,1
  r_table=190.100.60.0,24,130.175.1.2,1
  r_table=108.64.203.0,24,130.175.1.2,1
  r_table=203.96.251.0,24,130.175.1.2,1
  r_table=52.135.80.0,24,130.175.1.2,1
  r_table=82.46.91.0,24,130.175.1.2,1
  r_table=123.65.47.0,24,130.175.1.2,1
  r_table=201.34.64.0,24,130.175.1.2,1
  r_table=168.82.19.0,24,130.175.1.2,1
  r_table=134.86.70.0,24,130.175.1.2,1
  r_table=49.10.0.0,16,130.175.1.3,1
  r_table=140.183.234.0,24,130.175.1.3,1
  r_table=217.30.40.0,24,130.175.1.3,1
  r_table=148.48.0.0,16,130.175.1.3,1
  r_table=68.32.0.0,16,130.175.1.3,1
  r_table=12.110.0.0,16,130.175.1.3,1
  r_table=10.8.0.0,16,130.175.1.3,1
#Route to Proxy Board
  r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
  r_table=0.0.0.0,16,130.175.1.5,1
  Parms = {
  ..... name= Board 5 Outside Router 0
  /parms
/router

```

Figure 8: Outside Device Definition for Tap Board

Now that the structure of the config file has been explained it sounds easy to create a config file, right? Most people would be confused trying to write config files from scratch, in fact there are only a handful of people that have successfully written these config files. This is exactly where ISEConf comes in. The user no longer needs to worry about learning this syntax and making custom config files, ISEConf can do that for them. If you feel like learning more about the config file format or writing config files by hand, contact Dr. Jacobson to reference the ISEFlow config file format document.

Automatic ISEFlow Config Generator

ISEConf uses a relational database to keep track of all the details about backplanes, boards, devices, and interfaces. The users can simply add new backplanes, boards, devices, and interfaces to the configs by simply adding these through the web interface. The scripts create the configs automatically by pulling from this database and constructing the config file accordingly. This is mapped with the relations as seen in Figure 9. In this database all backplanes have a set of boards, each of these boards has a set of devices, and each device has a set of interfaces. These devices contain a doubly linked list that points to the next and previous device in a board, this can be used to trace through the devices.

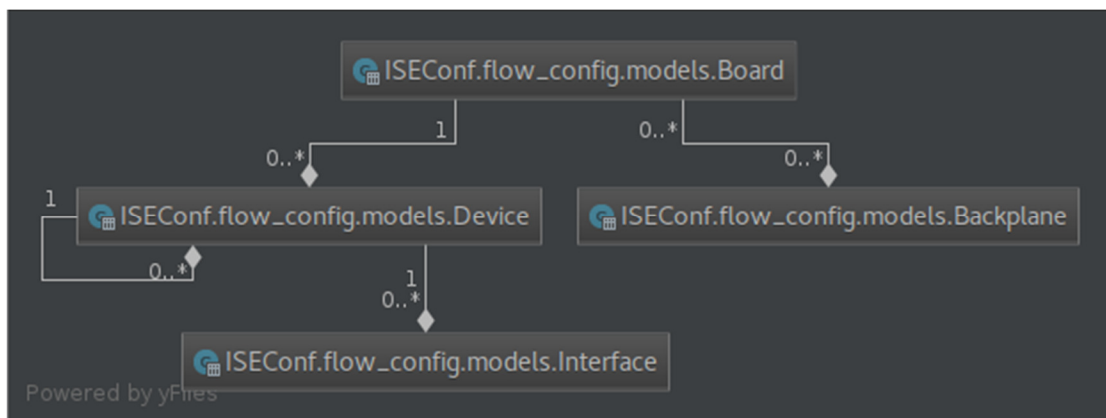


Figure 9: Database relations

The natural complexity of the config files is because we needed to store a lot of information about the devices and interfaces so we can distinguish how to add each line of the config files. The ISEFlow config files need to treat interfaces and devices differently depending on where they are located. As such there are three classifications of how the devices are placed in a board: inside, internal, and outside. The inside device is the one that is connected to the backplane and the outside device is one that has the interface_out to the IP ranges. Any devices between these two are considered internal devices. In the code each of these classifications of boards must be treated differently to properly structure the config file.

There are also four classifications on interfaces: two that ISEFlow uses and two more that are needed for the automatic generation of the config files. These classifications are interface_out, interface_to_next_device, interface_into_device, backplane_interface. These classifications allow knowledge of how to construct the definitions on each device. These allow the knowledge needed for all routing inside of the environment. Figure 10 shows the data that is stored in the database.


```

▼ flow_config_device
  id INTEGER
  device_type TEXT
  device_placement TEXT
  belongs_to_board_id INTEGER
  next_device_out_id INTEGER
  next_device_in_id INTEGER
  (id)
  #FAKE_flow_config_device_1 (next_device_in_id) → flow_config_device (id)
  #FAKE_flow_config_device_2 (next_device_out_id) → flow_config_device (id)
  #FAKE_flow_config_device_3 (belongs_to_board_id) → flow_config_board (id)
▼ flow_config_interface
  id INTEGER
  ip_address TEXT
  netmask INTEGER
  mac_address TEXT
  on_device_id INTEGER
  interface_type TEXT
  (id)
  #FAKE_flow_config_interface_1 (on_device_id) → flow_config_device (id)
  sqlite_autoindex_flow_config_interface_1 (ip_address) UNIQUE

```

Figure 10: Descriptions of database tables

Now that we know the technical details of how this works we can look at how it is practically used in ISEConf. The main idea of this Database is that it no longer requires someone that knows ISEFlow to write a config but anyone that understands networking. Adding Backplanes, Boards, Devices, and Interfaces are now as simple as using a GUI that verifies the user inputs. This allows any technically skilled person the ability to easily create a new config. Figure 11 Below shows the process for creating a new Device. This is simply filling out a webform where most options are dropdown selections to prevent invalid configurations.







Device type:	Router
Device placement:	Internal
Belongs to board:	Board #3  
Next device out:	Device type: Router position: Outside On Board: Board #3  
Next device in:	Device type: Router position: Inside On Board: Board #3  

Figure 11: Creating a new Device

Having user entered information in a database as well as information regarding what is inside ISEFlow makes it easy to have a script pull the data and do the heavy lifting of creating the config file. Appendix B is a config file that has been created using ISEConf. This shows how easy it could be for abstracting the config files away from the user to make it simpler to manage and run a fully customizable cyber security playground.

After this config file is created the ISEConf web application actually Takes it one step further to completely automate the process. It give the option to upload the config file, compile it and start running ISEFlow with the new configuration almost instantly! This is just one more example of how ISEConf works to simplify the user interactions between the end user and ISEFlow. This is one of the most effective ways for increasing the customizability of the ISERink playground environment.

ISEFlow Visualization

The visualization is currently implemented using the vis.js JavaScript library. vis.js has features specifically for creating graphs that can be interactive and customizable. The current implementation iterates through all the backplanes, boards, devices, and interfaces to show how the boards are connected, what devices are on each board and what the IP ranges are that can be connected to in the cyber security playground. This is a very basic implementation that shows the architecture of the environment. Figure 11 below shows what this visualization looks like. From this you can see how easy it is to tell what the testbed environment looks like. This is much simpler than what previously existed, where the only way to see the full network was to traceroute the environment or understand the config files.

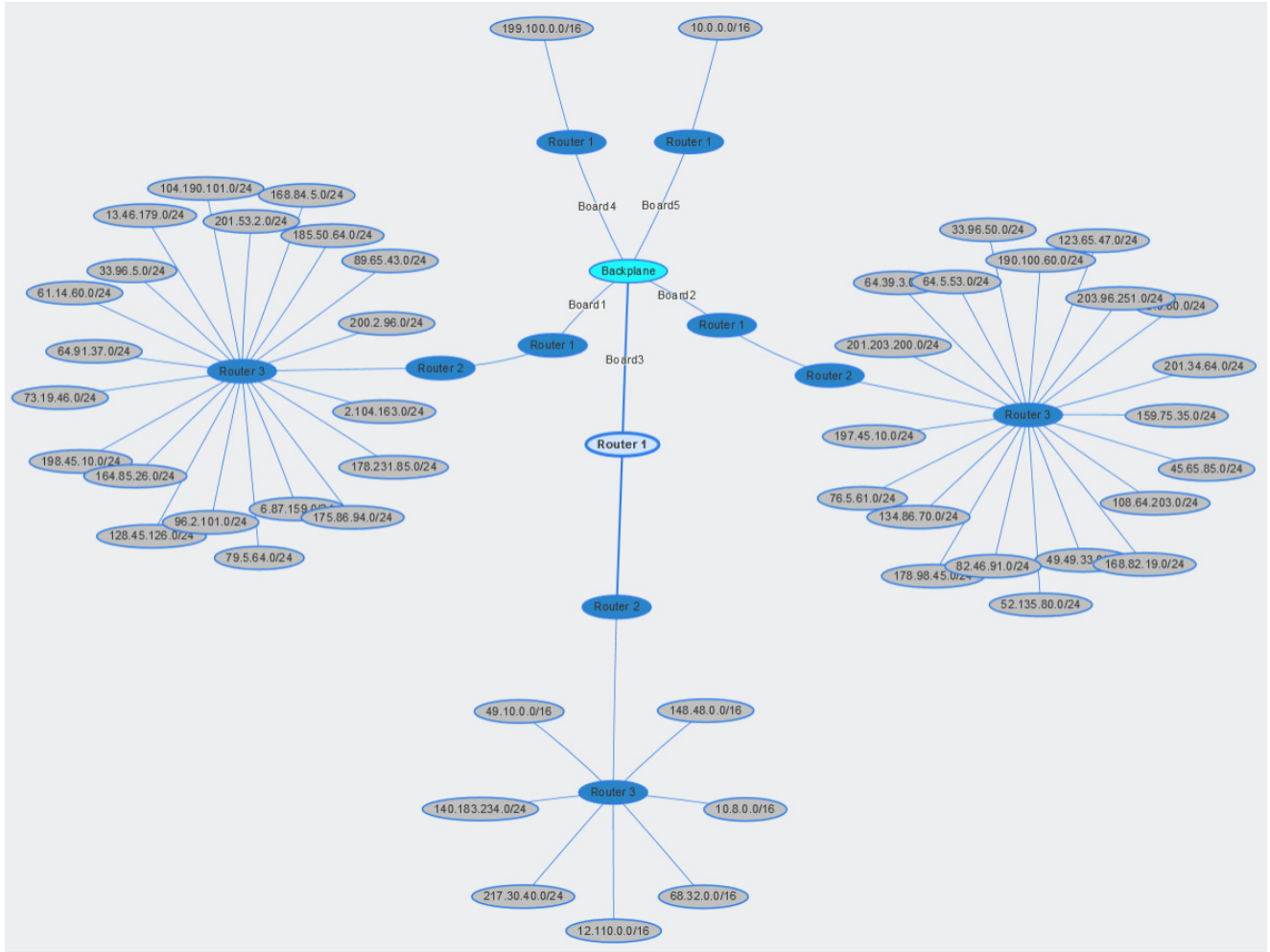


Figure 11: Visualization of the Configured Environment

ISEFlow Statistics

The statistics ISEConf is showing is using the data pulled directly from the running ISEFlow processes on each board. The ISEFlow processes keep track of packets that are sent between each IP range and within each board to create certain statistics. These statistics are being displayed directly and updating every thirty seconds to let people get a better idea of what is going on within the environment. This type of visualization and statistics are what people using ISERink would like to see for research and education. The statistics that ISEFlow is gathering

currently consists of packet flows at each board and the breakdown for each IP Range. This in combination with the Tap board can show a lot of interesting views of what is going on in the environment.

CHAPTER V

FUTURE WORKS

Improved Visualization

The implementation that is being proposed works very well for mapping out what the architecture of the environment looks like. However, being able to see real time packet flow in the environment would be an interesting future project. There are currently two possible ways of accomplishing this. The first is to combine the statistics that are being returned from ISEFlow and using this to update the map. This is a rather trivial approach and will not yield many details on what is going on besides packet count.

The second approach is to integrate directly with the traffic that the Tap board is spitting out with the map. This could give the ability to analyze what type of traffic is going through the network. This would probably yield better results since this will show all the packets that have made it through ISEFlow. ISEFlow will get some packets and simply drop them if it does not know how to handle this. One example is that ISEFlow in its current state will just drop IPv6 packets. If statistics are being pulled via ISEFlow these dropped packets may still be shown but running statistics on the TAP board will get the real flow of packets. This is an area that shows great potential for expansion and improvement, but can be approached from additional perspectives than discussed here.

Adding New Devices to ISEFlow

As mentioned earlier ISEFlow currently only supports the Router device, but there are plans for implementing new types of devices in ISEFlow. These other devices are Clouds, Links,

and Networks. These devices are designed to make the ISEFlow networking become more versatile. The addition of these devices has already been stubbed out in ISEConf so that when they are implemented in ISEFlow only small changes made in a few portions of the application will need to be made to make the new devices compatible with ISEConf.

Federation

One of the next major projects for the future is the federation of ISERinks. This will consist of gluing together multiple ISEFlow instances across several physical locations. This has been a difficult task up to this point since the config files had to be written by hand and in a federated ISERink all IP Ranges need to be unique on each ISERink. With ISEConf there is now a very easy way to automate the generation of config files and this could easily be adapted to have the ISEConf machines on each ISERink communicate and negotiate their config files accordingly. This would be done by having one of the ISERinks acting as a master for the federation that all other ISERinks would talk to. This communication would happen over a rest API built into the ISEConf application.

CHAPTER VI

SUMMARY AND CONCLUSION

In the future, ISEConf will be an asset to have in the ISERink environment. It will allow for the expansion of tools necessary for researchers and of new features necessary for future development. ISEConf will open a new direction for ISERink that will allow for more flexibility so that it can meet the needs of the end user. This could range from modeling networks, mock test environments for training of attack/response for cyber incidents, or educating the next generation of cyber security professionals. ISERink is an excellent tool for performing cyber security research and with new additions like ISEConf it will continue to be a great platform for years to come.

REFERENCES

Karstens, Nathan Lyle. "DeepFreeze: a Management Interface for ISEAGE." *Thesis / Dissertation ETD*, 2007.

Jacobson, Doug. "ISEFLOW Configuration File Format." 30 Aug. 2012.

Jacobson, Doug. "ISEAGE Testbed Overview." 2012.

Jacobson, Doug. "Design, Installation, and Configuration Document." 31 July 2012.

Jacobson, Doug, and Jeff Neel. "ISERink Installation Guide v1.3." 1 Jan. 2016.

[The Science of Cyber-Security Experimentation: The DETER Project](#). Terry Benzel. In Proceedings of the Annual Computer Security Applications Conference (ACSAC '11), Orlando, Florida, December 2011.

[The DETER Project: Advancing the Science of Cyber Security Experimentation and Test](#). Jelena Mirkovic, Terry V. Benzel, Ted Faber, Robert Braden, John T. Wroclawski, and Stephen Schwab. In Proceedings of the 2010 IEEE International Conference on Technologies for Homeland Security (HST '10), Waltham, Massachusetts, November 2010.

APPENDIX A OLD FLOW CONFIG

```

#begin board 1
board=1
router=0
type=outside
name=bd1rtr0
if_out=0,00:00:0c:01:00:00,33.96.5.254,33.96.5.0,24
if_out=1,00:00:0c:01:00:01,200.2.96.254,200.2.96.0,24
if_out=2,00:00:0c:01:00:02,79.5.64.254,79.5.64.0,24
if_out=3,00:00:0c:01:00:03,104.190.101.254,104.190.101.0,24
if_out=4,00:00:0c:01:00:04,6.87.159.254,6.87.159.0,24
if_out=5,00:00:0c:01:00:05,73.19.46.254,73.19.46.0,24
if_out=6,00:00:0c:01:00:06,89.65.43.254,89.65.43.0,24
if_out=7,00:00:0c:01:00:07,168.84.5.254,168.84.5.0,24
if_out=8,00:00:0c:01:00:08,175.86.94.254,175.86.94.0,24
if_out=9,00:00:0c:01:00:09,128.45.126.254,128.45.126.0,24
if_out=10,00:00:0c:01:00:10,96.2.101.254,96.2.101.0,24
if_out=11,00:00:0c:01:00:11,2.104.163.254,2.104.163.0,24
if_out=12,00:00:0c:01:00:12,201.53.2.254,201.53.2.0,24
if_out=13,00:00:0c:01:00:13,61.14.60.254,61.14.60.0,24
if_out=14,00:00:0c:01:00:14,198.45.10.254,198.45.10.0,24
if_out=15,00:00:0c:01:00:15,64.91.37.254,64.91.37.0,24
if_out=16,00:00:0c:01:00:16,13.46.179.254,13.46.179.0,24
if_out=17,00:00:0c:01:00:17,164.85.26.254,164.85.26.0,24
if_out=18,00:00:0c:01:00:18,185.50.64.254,185.50.64.0,24
if_out=19,00:00:0c:01:00:19,178.231.85.254,178.231.85.0,24
if=0,33.96.5.254,
if=1,200.2.96.254,
if=2,79.5.64.254,
if=3,104.190.101.254,
if=4,6.87.159.254,
if=5,73.19.46.254,
if=6,89.65.43.254,
if=7,168.84.5.254,
if=8,175.86.94.254,
if=9,128.45.126.254,
if=10,96.2.101.254,
if=11,2.104.163.254,
if=12,201.53.2.254,
if=13,61.14.60.254,
if=14,198.45.10.254,
if=15,64.91.37.254,
if=16,13.46.179.254,
if=17,164.85.26.254,
if=18,185.50.64.254,

```

```

if=19,178.231.85.254,
#Routes out to board 1 subnets
r_table=33.96.5.0,24,outside,33.96.5.254,1,0,0
r_table=200.2.96.0,24,outside,200.2.96.254,1,0,1
r_table=79.5.64.0,24,outside,79.5.64.254,1,0,2
r_table=104.190.101.0,24,outside,104.190.101.254,1,0,3
r_table=6.87.159.0,24,outside,6.87.159.254,1,0,4
r_table=73.19.46.0,24,outside,73.19.46.254,1,0,5
r_table=89.65.43.0,24,outside,89.65.43.254,1,0,6
r_table=168.84.5.0,24,outside,168.84.5.254,1,0,7
r_table=175.86.94.0,24,outside,175.86.94.254,1,0,8
r_table=128.45.126.0,24,outside,128.45.126.254,1,0,9
r_table=96.2.101.0,24,outside,96.2.101.254,1,0,10
r_table=2.104.163.0,24,outside,2.104.163.254,1,0,11
r_table=201.53.2.0,24,outside,201.53.2.254,1,0,12
r_table=61.14.60.0,24,outside,61.14.60.254,1,0,13
r_table=198.45.10.0,24,outside,198.45.10.254,1,0,14
r_table=64.91.37.0,24,outside,64.91.37.254,1,0,15
r_table=13.46.179.0,24,outside,13.46.179.254,1,0,16
r_table=164.85.26.0,24,outside,164.85.26.254,1,0,17
r_table=185.50.64.0,24,outside,185.50.64.254,1,0,18
r_table=178.231.85.0,24,outside,178.231.85.254,1,0,19
#default route
r_table=0.0.0.0,24,internal,130.1.1.100,1,1,0
router=1
type=internal
name=bd1rtr1
if=0,130.1.1.100,
if=1,130.1.1.101,
#internal routes to board 1 subnets
r_table=33.96.5.0,24,internal,33.96.5.254,1,0,0
r_table=200.2.96.0,24,internal,200.2.96.254,1,0,1
r_table=79.5.64.0,24,internal,79.5.64.254,1,0,2
r_table=104.190.101.0,24,internal,104.190.101.254,1,0,3
r_table=6.87.159.0,24,internal,6.87.159.254,1,0,4
r_table=73.19.46.0,24,internal,73.19.46.254,1,0,5
r_table=89.65.43.0,24,internal,89.65.43.254,1,0,6
r_table=168.84.5.0,24,internal,168.84.5.254,1,0,7
r_table=175.86.94.0,24,internal,175.86.94.254,1,0,8
r_table=128.45.126.0,24,internal,128.45.126.254,1,0,9
r_table=96.2.101.0,24,internal,96.2.101.254,1,0,10
r_table=2.104.163.0,24,internal,2.104.163.254,1,0,11
r_table=201.53.2.0,24,internal,201.53.2.254,1,0,12
r_table=61.14.60.0,24,internal,61.14.60.254,1,0,13
r_table=198.45.10.0,24,internal,198.45.10.254,1,0,14
r_table=64.91.37.0,24,internal,64.91.37.254,1,0,15

```

```

r_table=13.46.179.0,24,internal,13.46.179.254,1,0,16
r_table=164.85.26.0,24,internal,164.85.26.254,1,0,17
r_table=185.50.64.0,24,internal,185.50.64.254,1,0,18
r_table=178.231.85.0,24,internal,178.231.85.254,1,0,19
#default route
r_table=0.0.0.0,24,internal,130.1.2.100,1,2,0
router=2
type=inside
name=bd1rtr2
if=0,130.1.2.100,
if=1,130.1.2.101,
#internal routes to board 1 subnets
r_table=33.96.5.0,24,internal,130.1.1.101,1,1,1
r_table=200.2.96.0,24,internal,130.1.1.101,1,1,1
r_table=79.5.64.0,24,internal,130.1.1.101,1,1,1
r_table=104.190.101.0,24,internal,130.1.1.101,1,1,1
r_table=6.87.159.0,24,internal,130.1.1.101,1,1,1
r_table=73.19.46.0,24,internal,130.1.1.101,1,1,1
r_table=89.65.43.0,24,internal,130.1.1.101,1,1,1
r_table=168.84.5.0,24,internal,130.1.1.101,1,1,1
r_table=175.86.94.0,24,internal,130.1.1.101,1,1,1
r_table=128.45.126.0,24,internal,130.1.1.101,1,1,1
r_table=96.2.101.0,24,internal,130.1.1.101,1,1,1
r_table=2.104.163.0,24,internal,130.1.1.101,1,1,1
r_table=201.53.2.0,24,internal,130.1.1.101,1,1,1
r_table=61.14.60.0,24,internal,130.1.1.101,1,1,1
r_table=198.45.10.0,24,internal,130.1.1.101,1,1,1
r_table=64.91.37.0,24,internal,130.1.1.101,1,1,1
r_table=13.46.179.0,24,internal,130.1.1.101,1,1,1
r_table=164.85.26.0,24,internal,130.1.1.101,1,1,1
r_table=185.50.64.0,24,internal,130.1.1.101,1,1,1
r_table=178.231.85.0,24,internal,130.1.1.101,1,1,1
#default route to TAP board
r_table=0.0.0.0,24,inside,130.5.2.101,5,2,1
#end board 1
#begin board 2
board=2
router=0
type=outside
name=bd2rtr0
if_out=0,00:00:0c:02:00:00,64.39.3.254,64.39.3.0,24
if_out=1,00:00:0c:02:00:01,201.203.200.254,201.203.200.0,24
if_out=2,00:00:0c:02:00:02,64.5.53.254,64.5.53.0,24
if_out=3,00:00:0c:02:00:03,49.49.33.254,49.49.33.0,24
if_out=4,00:00:0c:02:00:04,33.96.50.254,33.96.50.0,24
if_out=5,00:00:0c:02:00:05,76.5.61.254,76.5.61.0,24

```

```

if_out=6,00:00:0c:02:00:06,45.65.85.254,45.65.85.0,24
if_out=7,00:00:0c:02:00:07,159.75.35.254,159.75.35.0,24
if_out=8,00:00:0c:02:00:08,178.98.45.254,178.98.45.0,24
if_out=9,00:00:0c:02:00:09,197.45.10.254,197.45.10.0,24
if_out=10,00:00:0c:02:00:10,5.0.80.254,5.0.80.0,24
if_out=11,00:00:0c:02:00:11,190.100.60.254,190.100.60.0,24
if_out=12,00:00:0c:02:00:12,108.64.203.254,108.64.203.0,24
if_out=13,00:00:0c:02:00:13,203.96.251.254,203.96.251.0,24
if_out=14,00:00:0c:02:00:14,52.135.80.254,52.135.80.0,24
if_out=15,00:00:0c:02:00:15,82.46.91.254,82.46.91.0,24
if_out=16,00:00:0c:02:00:16,123.65.47.254,123.65.47.0,24
if_out=17,00:00:0c:02:00:17,201.34.64.254,201.34.64.0,24
if_out=18,00:00:0c:02:00:18,168.82.19.254,168.82.19.0,24
if_out=19,00:00:0c:02:00:19,134.86.70.254,134.86.70.0,24
if=0,64.39.3.254,
if=1,201.203.200.254,
if=2,64.5.53.254,
if=3,49.49.33.254,
if=4,33.96.50.254,
if=5,76.5.61.254,
if=6,45.65.85.254,
if=7,159.75.35.254,
if=8,178.98.45.254,
if=9,197.45.10.254,
if=10,5.0.80.254,
if=11,190.100.60.254,
if=12,108.64.203.254,
if=13,203.96.251.254,
if=14,52.135.80.254,
if=15,82.46.91.254,
if=16,123.65.47.254,
if=17,201.34.64.254,
if=18,168.82.19.254,
if=19,134.86.70.254,
#Routes out to board 2 subnets
r_table=64.39.3.0,24,outside,64.39.3.254,2,0,0
r_table=201.203.200.0,24,outside,201.203.200.254,2,0,1
r_table=64.5.53.0,24,outside,64.5.53.254,2,0,2
r_table=49.49.33.0,24,outside,49.49.33.254,2,0,3
r_table=33.96.50.0,24,outside,33.96.50.254,2,0,4
r_table=76.5.61.0,24,outside,76.5.61.254,2,0,5
r_table=45.65.85.0,24,outside,45.65.85.254,2,0,6
r_table=159.75.35.0,24,outside,159.75.35.254,2,0,7
r_table=178.98.45.0,24,outside,178.98.45.254,2,0,8
r_table=197.45.10.0,24,outside,197.45.10.254,2,0,9
r_table=5.0.80.0,24,outside,5.0.80.254,2,0,10

```

```

r_table=190.100.60.0,24,outside,190.100.60.254,2,0,11
r_table=108.64.203.0,24,outside,108.64.203.254,2,0,12
r_table=203.96.251.0,24,outside,203.96.251.254,2,0,13
r_table=52.135.80.0,24,outside,52.135.80.254,2,0,14
r_table=82.46.91.0,24,outside,82.46.91.254,2,0,15
r_table=123.65.47.0,24,outside,123.65.47.254,2,0,16
r_table=201.34.64.0,24,outside,201.34.64.254,2,0,17
r_table=168.82.19.0,24,outside,168.82.19.254,2,0,18
r_table=134.86.70.0,24,outside,134.86.70.254,2,0,19
#default route
r_table=0.0.0.0,24,internal,130.2.1.100,2,1,0
router=1
type=internal
name=bd2rtr1
if=0,130.2.1.100,
if=1,130.2.1.101,
#internal routes to board 2 subnets
r_table=64.39.3.0,24,internal,64.39.3.254,2,0,0
r_table=201.203.200.0,24,internal,201.203.200.254,2,0,1
r_table=64.5.53.0,24,internal,64.5.53.254,2,0,2
r_table=49.49.33.0,24,internal,49.49.33.254,2,0,3
r_table=33.96.50.0,24,internal,33.96.50.254,2,0,4
r_table=76.5.61.0,24,internal,76.5.61.254,2,0,5
r_table=45.65.85.0,24,internal,45.65.85.254,2,0,6
r_table=159.75.35.0,24,internal,159.75.35.254,2,0,7
r_table=178.98.45.0,24,internal,178.98.45.254,2,0,8
r_table=197.45.10.0,24,internal,197.45.10.254,2,0,9
r_table=5.0.80.0,24,internal,5.0.80.254,2,0,10
r_table=190.100.60.0,24,internal,190.100.60.254,2,0,11
r_table=108.64.203.0,24,internal,108.64.203.254,2,0,12
r_table=203.96.251.0,24,internal,203.96.251.254,2,0,13
r_table=52.135.80.0,24,internal,52.135.80.254,2,0,14
r_table=82.46.91.0,24,internal,82.46.91.254,2,0,15
r_table=123.65.47.0,24,internal,123.65.47.254,2,0,16
r_table=201.34.64.0,24,internal,201.34.64.254,2,0,17
r_table=168.82.19.0,24,internal,168.82.19.254,2,0,18
r_table=134.86.70.0,24,internal,134.86.70.254,2,0,19
#default route
r_table=0.0.0.0,24,internal,130.2.2.100,2,2,0
router=2
type=inside
name=bd2rtr2
if=0,130.2.2.100,
if=1,130.2.2.101,
#internal routes to board 2 subnets
r_table=64.39.3.0,24,internal,130.2.1.101,2,1,1

```

```

r_table=201.203.200.0,24,internal,130.2.1.101,2,1,1
r_table=64.5.53.0,24,internal,130.2.1.101,2,1,1
r_table=49.49.33.0,24,internal,130.2.1.101,2,1,1
r_table=33.96.50.0,24,internal,130.2.1.101,2,1,1
r_table=76.5.61.0,24,internal,130.2.1.101,2,1,1
r_table=45.65.85.0,24,internal,130.2.1.101,2,1,1
r_table=159.75.35.0,24,internal,130.2.1.101,2,1,1
r_table=178.98.45.0,24,internal,130.2.1.101,2,1,1
r_table=197.45.10.0,24,internal,130.2.1.101,2,1,1
r_table=5.0.80.0,24,internal,130.2.1.101,2,1,1
r_table=190.100.60.0,24,internal,130.2.1.101,2,1,1
r_table=108.64.203.0,24,internal,130.2.1.101,2,1,1
r_table=203.96.251.0,24,internal,130.2.1.101,2,1,1
r_table=52.135.80.0,24,internal,130.2.1.101,2,1,1
r_table=82.46.91.0,24,internal,130.2.1.101,2,1,1
r_table=123.65.47.0,24,internal,130.2.1.101,2,1,1
r_table=201.34.64.0,24,internal,130.2.1.101,2,1,1
r_table=168.82.19.0,24,internal,130.2.1.101,2,1,1
r_table=134.86.70.0,24,internal,130.2.1.101,2,1,1
#default route to TAP board
r_table=0.0.0.0,24,inside,130.5.2.101,5,2,1
#end board 2
#begin board 3
board=3
router=0
type=outside
name=bd3rtr0
if_out=0,00:00:0c:03:00:00,49.10.254.254,49.10.0.0,16
if_out=1,00:00:0c:03:00:01,140.183.234.254,140.183.234.0,24
if_out=2,00:00:0c:03:00:02,217.30.40.254,217.30.40.0,24
if_out=3,00:00:0c:03:00:03,148.48.254.254,148.48.0.0,16
if_out=4,00:00:0c:03:00:04,68.32.254.254,68.32.0.0,16
if_out=5,00:00:0c:03:00:05,12.110.254.254,12.110.0.0,16
if_out=6,00:00:0c:03:00:06,10.8.254.254,10.8.0.0,16
if=0,49.10.254.254,
if=1,140.183.234.254,
if=2,217.30.40.254,
if=3,148.48.254.254,
if=4,68.32.254.254,
if=5,12.110.254.254,
if=6,10.8.254.254,
#Routes out to board 3 subnets
r_table=49.10.0.0,16,outside,49.10.254.254,3,0,0
r_table=140.183.234.0,24,outside,140.183.234.254,3,0,1
r_table=217.30.40.0,24,outside,217.30.40.254,3,0,2
r_table=148.48.0.0,16,outside,148.48.254.254,3,0,3

```

```

r_table=68.32.0.0,16,outside,68.32.254.254,3,0,4
r_table=12.110.0.0,16,outside,12.110.254.254,3,0,5
r_table=10.8.0.0,16,outside,10.8.254.254,3,0,6
#default route
r_table=0.0.0.0,24,internal,130.3.1.100,3,1,0
router=1
type=internal
name=bd3rtr1
if=0,130.3.1.100,
if=1,130.3.1.101,
#internal routes to board 3 subnets
r_table=49.10.0.0,16,internal,49.10.254.254,3,0,0
r_table=140.183.234.0,24,internal,140.183.234.254,3,0,1
r_table=217.30.40.0,24,internal,217.30.40.254,3,0,2
r_table=148.48.0.0,16,internal,148.48.254.254,3,0,3
r_table=68.32.0.0,16,internal,68.32.254.254,3,0,4
r_table=12.110.0.0,16,internal,12.110.254.254,3,0,5
r_table=10.8.0.0,16,internal,10.8.254.254,3,0,6
#default route
r_table=0.0.0.0,24,internal,130.3.2.100,3,2,0
router=2
type=inside
name=bd3rtr2
if=0,130.3.2.100,
if=1,130.3.2.101,
#internal routes to board 3 subnets
r_table=49.10.0.0,16,internal,130.3.1.101,3,1,1
r_table=140.183.234.0,24,internal,130.3.1.101,3,1,1
r_table=217.30.40.0,24,internal,130.3.1.101,3,1,1
r_table=148.48.0.0,16,internal,130.3.1.101,3,1,1
r_table=68.32.0.0,16,internal,130.3.1.101,3,1,1
r_table=12.110.0.0,16,internal,130.3.1.101,3,1,1
r_table=10.8.0.0,16,internal,130.3.1.101,3,1,1
#default route to TAP board
r_table=0.0.0.0,24,inside,130.5.2.101,5,2,1
#end board 3
#begin board 4
board=4
router=0
type=outside
name=bd4rtr0
if_out=0,00:00:0c:04:00:00,199.100.254.254,199.100.0.0,16
if=0,199.100.254.254,
#Routes out to board 4 subnets
r_table=199.100.0.0,16,outside,199.100.254.254,4,0,0
#default route

```



```

r_table=0.0.0.0,24,internal,130.4.1.100,4,1,0
router=1
type=internal
name=bd4rtr1
if=0,130.4.1.100,
if=1,130.4.1.101,
#internal routes to board 4 subnets
r_table=199.100.0.0,16,internal,199.100.254.254,4,0,0
#default route
r_table=0.0.0.0,24,internal,130.4.2.100,4,2,0
router=2
type=inside
name=bd4rtr2
if=0,130.4.2.100,
if=1,130.4.2.101,
#internal routes to board 4 subnets
r_table=199.100.0.0,16,internal,130.4.1.101,4,1,1
#default route to TAP board
r_table=0.0.0.0,24,inside,130.5.2.101,5,2,1
#end board 4
#begin board 5
board=5
tap=1
router=0
type=outside
name=bd5rtr0
if_out=0,00:00:0c:05:00:00,10.254.254.254,10.254.0.0,16
if=0,10.254.254.254,
#default route
r_table=0.0.0.0,24,outside,10.254.0.0,5,0,0
router=1
type=internal
name=bd5rtr1
if=0,130.5.1.100,
if=1,130.5.1.101,
#default route to dump traffic
r_table=0.0.0.0,24,internal,10.254.254.254,5,0,0
router=2
type=inside
name=bd5rtr2
if=0,130.5.2.100,
if=1,130.5.2.101,
#routes to board 1
r_table=33.96.5.0,24,inside,130.1.2.101,1,2,1
r_table=200.2.96.0,24,inside,130.1.2.101,1,2,1
r_table=79.5.64.0,24,inside,130.1.2.101,1,2,1

```

```

r_table=104.190.101.0,24,inside,130.1.2.101,1,2,1
r_table=6.87.159.0,24,inside,130.1.2.101,1,2,1
r_table=73.19.46.0,24,inside,130.1.2.101,1,2,1
r_table=89.65.43.0,24,inside,130.1.2.101,1,2,1
r_table=168.84.5.0,24,inside,130.1.2.101,1,2,1
r_table=175.86.94.0,24,inside,130.1.2.101,1,2,1
r_table=128.45.126.0,24,inside,130.1.2.101,1,2,1
r_table=96.2.101.0,24,inside,130.1.2.101,1,2,1
r_table=2.104.163.0,24,inside,130.1.2.101,1,2,1
r_table=201.53.2.0,24,inside,130.1.2.101,1,2,1
r_table=61.14.60.0,24,inside,130.1.2.101,1,2,1
r_table=198.45.10.0,24,inside,130.1.2.101,1,2,1
r_table=64.91.37.0,24,inside,130.1.2.101,1,2,1
r_table=13.46.179.0,24,inside,130.1.2.101,1,2,1
r_table=164.85.26.0,24,inside,130.1.2.101,1,2,1
r_table=185.50.64.0,24,inside,130.1.2.101,1,2,1
r_table=178.231.85.0,24,inside,130.1.2.101,1,2,1
#routes to board 2
r_table=64.39.3.0,24,inside,130.2.2.101,2,2,1
r_table=201.203.200.0,24,inside,130.2.2.101,2,2,1
r_table=64.5.53.0,24,inside,130.2.2.101,2,2,1
r_table=49.49.33.0,24,inside,130.2.2.101,2,2,1
r_table=33.96.50.0,24,inside,130.2.2.101,2,2,1
r_table=76.5.61.0,24,inside,130.2.2.101,2,2,1
r_table=45.65.85.0,24,inside,130.2.2.101,2,2,1
r_table=159.75.35.0,24,inside,130.2.2.101,2,2,1
r_table=178.98.45.0,24,inside,130.2.2.101,2,2,1
r_table=197.45.10.0,24,inside,130.2.2.101,2,2,1
r_table=5.0.80.0,24,inside,130.2.2.101,2,2,1
r_table=190.100.60.0,24,inside,130.2.2.101,2,2,1
r_table=108.64.203.0,24,inside,130.2.2.101,2,2,1
r_table=203.96.251.0,24,inside,130.2.2.101,2,2,1
r_table=52.135.80.0,24,inside,130.2.2.101,2,2,1
r_table=82.46.91.0,24,inside,130.2.2.101,2,2,1
r_table=123.65.47.0,24,inside,130.2.2.101,2,2,1
r_table=201.34.64.0,24,inside,130.2.2.101,2,2,1
r_table=168.82.19.0,24,inside,130.2.2.101,2,2,1
r_table=134.86.70.0,24,inside,130.2.2.101,2,2,1
#routes to board 3
r_table=49.10.0.0,16,inside,130.3.2.101,3,2,1
r_table=140.183.234.0,24,inside,130.3.2.101,3,2,1
r_table=217.30.40.0,24,inside,130.3.2.101,3,2,1
r_table=148.48.0.0,16,inside,130.3.2.101,3,2,1
r_table=68.32.0.0,16,inside,130.3.2.101,3,2,1
r_table=12.110.0.0,16,inside,130.3.2.101,3,2,1
#routes to board 4

```

```
r_table=199.100.0.0,16,inside,130.4.2.101,4,2,1  
#default route  
r_table=0.0.0.0,24,inside,130.5.1.101,5,1,1  
#end board 5
```

APPENDIX B NEW FLOW CONFIG

```
#####
#                                                                 #
#           Config file generated By ISEConf web generator       #
#                                                                 #
#####
VER=1,0
##### Global Definitions #####
Globals= {
  BPnet=0,130.175.1.0,24
    board=1,3,1,130.175.1.1
    board=2,3,1,130.175.1.2
    board=3,3,1,130.175.1.3
    board=4,1,1,130.175.1.4
    board=5,1,1,130.175.1.5
  DLINK=IP
  Parms = {
    name=generic_link
  /parms

  GLINK=1,IP
  Parms = {
    name=IP link with loss
    # 1 percent loss
    loss=1.0
  /parms
/global
##### Board Definitions #####

#####Board 1#####
board=1
connections={
  R0,0 => O
  R0,1 => O
  R0,2 => O
  R0,3 => O
  R0,4 => O
  R0,5 => O
  R0,6 => O
  R0,7 => O
  R0,8 => O
  R0,9 => O
  R0,10 => O
  R0,11 => O
}
```

```

R0,12 => O
R0,13 => O
R0,14 => O
R0,15 => O
R0,16 => O
R0,17 => O
R0,18 => O
R0,19 => O
R0,20 => R1, 0, IP
R1,1 => R2,0 , L
R2,1 => B0, G, IP
/connection
device=router,0
if_out=0,33.96.5.0,24,00:00:0c:01:00:00
if_out=1,200.2.96.0,24,00:00:0c:01:00:01
if_out=2,79.5.64.0,24,00:00:0c:01:00:02
if_out=3,104.190.101.0,24,00:00:0c:01:00:03
if_out=4,6.87.159.0,24,00:00:0c:01:00:04
if_out=5,73.19.46.0,24,00:00:0c:01:00:05
if_out=6,89.65.43.0,24,00:00:0c:01:00:06
if_out=7,168.84.5.0,24,00:00:0c:01:00:07
if_out=8,175.86.94.0,24,00:00:0c:01:00:08
if_out=9,128.45.126.0,24,00:00:0c:01:00:09
if_out=10,96.2.101.0,24,00:00:0c:01:00:10
if_out=11,2.104.163.0,24,00:00:0c:01:00:11
if_out=12,201.53.2.0,24,00:00:0c:01:00:12
if_out=13,61.14.60.0,24,00:00:0c:01:00:13
if_out=14,198.45.10.0,24,00:00:0c:01:00:14
if_out=15,64.91.37.0,24,00:00:0c:01:00:15
if_out=16,13.46.179.0,24,00:00:0c:01:00:16
if_out=17,164.85.26.0,24,00:00:0c:01:00:17
if_out=18,185.50.64.0,24,00:00:0c:01:00:18
if_out=19,178.231.85.0,24,00:00:0c:01:00:19
if=20,130.170.1.100,24
##### R Tables #####
r_table=33.96.5.0,24,33.96.5.254,0
r_table=200.2.96.0,24,200.2.96.254,1
r_table=79.5.64.0,24,79.5.64.254,2
r_table=104.190.101.0,24,104.190.101.254,3
r_table=6.87.159.0,24,6.87.159.254,4
r_table=73.19.46.0,24,73.19.46.254,5
r_table=89.65.43.0,24,89.65.43.254,6
r_table=168.84.5.0,24,168.84.5.254,7
r_table=175.86.94.0,24,175.86.94.254,8
r_table=128.45.126.0,24,128.45.126.254,9
r_table=96.2.101.0,24,96.2.101.254,10

```

```

r_table=2.104.163.0,24,2.104.163.254,11
r_table=201.53.2.0,24,201.53.2.254,12
r_table=61.14.60.0,24,61.14.60.254,13
r_table=198.45.10.0,24,198.45.10.254,14
r_table=64.91.37.0,24,64.91.37.254,15
r_table=13.46.179.0,24,13.46.179.254,16
r_table=164.85.26.0,24,164.85.26.254,17
r_table=185.50.64.0,24,185.50.64.254,18
r_table=178.231.85.0,24,178.231.85.254,19
# To Inside
r_table=0.0.0.0,24,130.170.1.254,20
Parms = {
    name= Board 1 Outside Router 0
}/parms
/router
device=router,1
if=0,130.170.1.254,24
if=1,131.10.1.100,24
    ##### R Tables #####
r_table=33.96.5.0,24,131.10.1.100,0
r_table=200.2.96.0,24,131.10.1.100,0
r_table=79.5.64.0,24,131.10.1.100,0
r_table=104.190.101.0,24,131.10.1.100,0
r_table=6.87.159.0,24,131.10.1.100,0
r_table=73.19.46.0,24,131.10.1.100,0
r_table=89.65.43.0,24,131.10.1.100,0
r_table=168.84.5.0,24,131.10.1.100,0
r_table=175.86.94.0,24,131.10.1.100,0
r_table=128.45.126.0,24,131.10.1.100,0
r_table=96.2.101.0,24,131.10.1.100,0
r_table=2.104.163.0,24,131.10.1.100,0
r_table=201.53.2.0,24,131.10.1.100,0
r_table=61.14.60.0,24,131.10.1.100,0
r_table=198.45.10.0,24,131.10.1.100,0
r_table=64.91.37.0,24,131.10.1.100,0
r_table=13.46.179.0,24,131.10.1.100,0
r_table=164.85.26.0,24,131.10.1.100,0
r_table=185.50.64.0,24,131.10.1.100,0
r_table=178.231.85.0,24,131.10.1.100,0
# To Inside
r_table=0.0.0.0,24,131.10.1.254,1
Parms = {
    name= Board 1 Internal Router 1
}/parms
/router
device=router,2

```

```

if=0,131.10.1.254,24
if=1,130.175.1.1,24
##### R Tables #####
r_table=33.96.5.0,24,131.10.1.100,0
r_table=200.2.96.0,24,131.10.1.100,0
r_table=79.5.64.0,24,131.10.1.100,0
r_table=104.190.101.0,24,131.10.1.100,0
r_table=6.87.159.0,24,131.10.1.100,0
r_table=73.19.46.0,24,131.10.1.100,0
r_table=89.65.43.0,24,131.10.1.100,0
r_table=168.84.5.0,24,131.10.1.100,0
r_table=175.86.94.0,24,131.10.1.100,0
r_table=128.45.126.0,24,131.10.1.100,0
r_table=96.2.101.0,24,131.10.1.100,0
r_table=2.104.163.0,24,131.10.1.100,0
r_table=201.53.2.0,24,131.10.1.100,0
r_table=61.14.60.0,24,131.10.1.100,0
r_table=198.45.10.0,24,131.10.1.100,0
r_table=64.91.37.0,24,131.10.1.100,0
r_table=13.46.179.0,24,131.10.1.100,0
r_table=164.85.26.0,24,131.10.1.100,0
r_table=185.50.64.0,24,131.10.1.100,0
r_table=178.231.85.0,24,131.10.1.100,0
#Route to Proxy Board
r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
r_table=0.0.0.0,16,130.175.1.5,1
Parms = {
    name= Board 1 Inside Router 2
}
/parms
/router
/board
#####Board 2#####
board=2
connections={
R0,0 => O
R0,1 => O
R0,2 => O
R0,3 => O
R0,4 => O
R0,5 => O
R0,6 => O
R0,7 => O
R0,8 => O
R0,9 => O
R0,10 => O

```

```

R0,11 => O
R0,12 => O
R0,13 => O
R0,14 => O
R0,15 => O
R0,16 => O
R0,17 => O
R0,18 => O
R0,19 => O
R0,20 => R1, 0, IP
R1,1 => R2,0 , L
R2,1 => B0, G, IP
/connection
device=router,0
if_out=0,64.39.3.0,24,00:00:0c:02:00:00
if_out=1,201.203.200.0,24,00:00:0c:02:00:01
if_out=2,64.5.53.0,24,00:00:0c:02:00:02
if_out=3,49.49.33.0,24,00:00:0c:02:00:03
if_out=4,33.96.50.0,24,00:00:0c:02:00:04
if_out=5,76.5.61.0,24,00:00:0c:02:00:05
if_out=6,45.65.85.0,24,00:00:0c:02:00:06
if_out=7,159.75.35.0,24,00:00:0c:02:00:07
if_out=8,178.98.45.0,24,00:00:0c:02:00:08
if_out=9,197.45.10.0,24,00:00:0c:02:00:09
if_out=10,5.0.80.0,24,00:00:0c:02:00:10
if_out=11,190.100.60.0,24,00:00:0c:02:00:11
if_out=12,108.64.203.0,24,00:00:0c:02:00:12
if_out=13,203.96.251.0,24,00:00:0c:02:00:13
if_out=14,52.135.80.0,24,00:00:0c:02:00:14
if_out=15,82.46.91.0,24,00:00:0c:02:00:15
if_out=16,123.65.47.0,24,00:00:0c:02:00:16
if_out=17,201.34.64.0,24,00:00:0c:02:00:17
if_out=18,168.82.19.0,24,00:00:0c:02:00:18
if_out=19,134.86.70.0,24,00:00:0c:02:00:19
if=20,130.170.2.100,24
##### R Tables #####
r_table=64.39.3.0,24,64.39.3.254,0
r_table=201.203.200.0,24,201.203.200.254,1
r_table=64.5.53.0,24,64.5.53.254,2
r_table=49.49.33.0,24,49.49.33.254,3
r_table=33.96.50.0,24,33.96.50.254,4
r_table=76.5.61.0,24,76.5.61.254,5
r_table=45.65.85.0,24,45.65.85.254,6
r_table=159.75.35.0,24,159.75.35.254,7
r_table=178.98.45.0,24,178.98.45.254,8
r_table=197.45.10.0,24,197.45.10.254,9

```



```

r_table=5.0.80.0,24,5.0.80.254,10
r_table=190.100.60.0,24,190.100.60.254,11
r_table=108.64.203.0,24,108.64.203.254,12
r_table=203.96.251.0,24,203.96.251.254,13
r_table=52.135.80.0,24,52.135.80.254,14
r_table=82.46.91.0,24,82.46.91.254,15
r_table=123.65.47.0,24,123.65.47.254,16
r_table=201.34.64.0,24,201.34.64.254,17
r_table=168.82.19.0,24,168.82.19.254,18
r_table=134.86.70.0,24,134.86.70.254,19
# To Inside
r_table=0.0.0.0,24,130.170.2.254,20
Parms = {
    name= Board 2 Outside Router 0
}
/parms
/router
device=router,1
if=0,130.170.2.254,24
if=1,131.10.2.100,24
##### R Tables #####
r_table=64.39.3.0,24,131.10.2.100,0
r_table=201.203.200.0,24,131.10.2.100,0
r_table=64.5.53.0,24,131.10.2.100,0
r_table=49.49.33.0,24,131.10.2.100,0
r_table=33.96.50.0,24,131.10.2.100,0
r_table=76.5.61.0,24,131.10.2.100,0
r_table=45.65.85.0,24,131.10.2.100,0
r_table=159.75.35.0,24,131.10.2.100,0
r_table=178.98.45.0,24,131.10.2.100,0
r_table=197.45.10.0,24,131.10.2.100,0
r_table=5.0.80.0,24,131.10.2.100,0
r_table=190.100.60.0,24,131.10.2.100,0
r_table=108.64.203.0,24,131.10.2.100,0
r_table=203.96.251.0,24,131.10.2.100,0
r_table=52.135.80.0,24,131.10.2.100,0
r_table=82.46.91.0,24,131.10.2.100,0
r_table=123.65.47.0,24,131.10.2.100,0
r_table=201.34.64.0,24,131.10.2.100,0
r_table=168.82.19.0,24,131.10.2.100,0
r_table=134.86.70.0,24,131.10.2.100,0
# To Inside
r_table=0.0.0.0,24,131.10.2.254,1
Parms = {
    name= Board 2 Internal Router 1
}
/parms
/router

```

```

device=router,2
  if=0,131.10.2.254,24
  if=1,130.175.1.2,24
      ##### R Tables #####
r_table=64.39.3.0,24,131.10.2.100,0
r_table=201.203.200.0,24,131.10.2.100,0
r_table=64.5.53.0,24,131.10.2.100,0
r_table=49.49.33.0,24,131.10.2.100,0
r_table=33.96.50.0,24,131.10.2.100,0
r_table=76.5.61.0,24,131.10.2.100,0
r_table=45.65.85.0,24,131.10.2.100,0
r_table=159.75.35.0,24,131.10.2.100,0
r_table=178.98.45.0,24,131.10.2.100,0
r_table=197.45.10.0,24,131.10.2.100,0
r_table=5.0.80.0,24,131.10.2.100,0
r_table=190.100.60.0,24,131.10.2.100,0
r_table=108.64.203.0,24,131.10.2.100,0
r_table=203.96.251.0,24,131.10.2.100,0
r_table=52.135.80.0,24,131.10.2.100,0
r_table=82.46.91.0,24,131.10.2.100,0
r_table=123.65.47.0,24,131.10.2.100,0
r_table=201.34.64.0,24,131.10.2.100,0
r_table=168.82.19.0,24,131.10.2.100,0
r_table=134.86.70.0,24,131.10.2.100,0
#Route to Proxy Board
  r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
  r_table=0.0.0.0,16,130.175.1.5,1
  Params = {
    name= Board 2 Inside Router 2
  }/params
/router
/board
#####Board 3#####
board=3
connections={
  R0,0 => O
  R0,1 => O
  R0,2 => O
  R0,3 => O
  R0,4 => O
  R0,5 => O
  R0,6 => O
  R0,7 => R1, 0, IP
  R1,1 => R2,0 , L
  R2,1 => B0, G, IP

```

```

/connection
device=router,0
  if_out=0,49.10.0.0,16,00:00:0c:03:00:00
  if_out=1,140.183.234.0,24,00:00:0c:03:00:01
  if_out=2,217.30.40.0,24,00:00:0c:03:00:02
  if_out=3,148.48.0.0,16,00:00:0c:03:00:03
  if_out=4,68.32.0.0,16,00:00:0c:03:00:04
  if_out=5,12.110.0.0,16,00:00:0c:03:00:05
  if_out=6,10.8.0.0,16,00:00:0c:03:00:06
  if=7,130.170.3.100,24
      ##### R Tables #####
  r_table=49.10.0.0,16,49.10.255.254,0
  r_table=140.183.234.0,24,140.183.234.254,1
  r_table=217.30.40.0,24,217.30.40.254,2
  r_table=148.48.0.0,16,148.48.255.254,3
  r_table=68.32.0.0,16,68.32.255.254,4
  r_table=12.110.0.0,16,12.110.255.254,5
  r_table=10.8.0.0,16,10.8.255.254,6
# To Inside
  r_table=0.0.0.0,24,130.170.3.254,7
  Parms = {
    name= Board 3 Outside Router 0
  }
/parms
/router
device=router,1
  if=0,130.170.3.254,24
  if=1,131.10.3.100,24
      ##### R Tables #####
  r_table=49.10.0.0,16,131.10.3.100,0
  r_table=140.183.234.0,24,131.10.3.100,0
  r_table=217.30.40.0,24,131.10.3.100,0
  r_table=148.48.0.0,16,131.10.3.100,0
  r_table=68.32.0.0,16,131.10.3.100,0
  r_table=12.110.0.0,16,131.10.3.100,0
  r_table=10.8.0.0,16,131.10.3.100,0
# To Inside
  r_table=0.0.0.0,24,131.10.3.254,1
  Parms = {
    name= Board 3 Internal Router 1
  }
/parms
/router
device=router,2
  if=0,131.10.3.254,24
  if=1,130.175.1.3,24
      ##### R Tables #####
  r_table=49.10.0.0,16,131.10.3.100,0

```

```

r_table=140.183.234.0,24,131.10.3.100,0
r_table=217.30.40.0,24,131.10.3.100,0
r_table=148.48.0.0,16,131.10.3.100,0
r_table=68.32.0.0,16,131.10.3.100,0
r_table=12.110.0.0,16,131.10.3.100,0
r_table=10.8.0.0,16,131.10.3.100,0
#Route to Proxy Board
r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
r_table=0.0.0.0,16,130.175.1.5,1
Parms = {
    name= Board 3 Inside Router 2
}
/parms
/router
/board
#####Board 4#####
board=4
connections={
    R0,0 => O
    R0,1 => B0, G, IP
}
/connections
device=router,0
if_out=0,199.100.0.0,16,00:00:0c:04:00:00
if=1,130.175.1.4,24
##### R Tables #####
r_table=199.100.0.0,16,199.100.255.254,0
#Route to Internal Boards
r_table=33.96.5.0,24,130.175.1.1,1
r_table=200.2.96.0,24,130.175.1.1,1
r_table=79.5.64.0,24,130.175.1.1,1
r_table=104.190.101.0,24,130.175.1.1,1
r_table=6.87.159.0,24,130.175.1.1,1
r_table=73.19.46.0,24,130.175.1.1,1
r_table=89.65.43.0,24,130.175.1.1,1
r_table=168.84.5.0,24,130.175.1.1,1
r_table=175.86.94.0,24,130.175.1.1,1
r_table=128.45.126.0,24,130.175.1.1,1
r_table=96.2.101.0,24,130.175.1.1,1
r_table=2.104.163.0,24,130.175.1.1,1
r_table=201.53.2.0,24,130.175.1.1,1
r_table=61.14.60.0,24,130.175.1.1,1
r_table=198.45.10.0,24,130.175.1.1,1
r_table=64.91.37.0,24,130.175.1.1,1
r_table=13.46.179.0,24,130.175.1.1,1
r_table=164.85.26.0,24,130.175.1.1,1
r_table=185.50.64.0,24,130.175.1.1,1

```

```

r_table=178.231.85.0,24,130.175.1.1,1
r_table=64.39.3.0,24,130.175.1.2,1
r_table=201.203.200.0,24,130.175.1.2,1
r_table=64.5.53.0,24,130.175.1.2,1
r_table=49.49.33.0,24,130.175.1.2,1
r_table=33.96.50.0,24,130.175.1.2,1
r_table=76.5.61.0,24,130.175.1.2,1
r_table=45.65.85.0,24,130.175.1.2,1
r_table=159.75.35.0,24,130.175.1.2,1
r_table=178.98.45.0,24,130.175.1.2,1
r_table=197.45.10.0,24,130.175.1.2,1
r_table=5.0.80.0,24,130.175.1.2,1
r_table=190.100.60.0,24,130.175.1.2,1
r_table=108.64.203.0,24,130.175.1.2,1
r_table=203.96.251.0,24,130.175.1.2,1
r_table=52.135.80.0,24,130.175.1.2,1
r_table=82.46.91.0,24,130.175.1.2,1
r_table=123.65.47.0,24,130.175.1.2,1
r_table=201.34.64.0,24,130.175.1.2,1
r_table=168.82.19.0,24,130.175.1.2,1
r_table=134.86.70.0,24,130.175.1.2,1
r_table=49.10.0.0,16,130.175.1.3,1
r_table=140.183.234.0,24,130.175.1.3,1
r_table=217.30.40.0,24,130.175.1.3,1
r_table=148.48.0.0,16,130.175.1.3,1
r_table=68.32.0.0,16,130.175.1.3,1
r_table=12.110.0.0,16,130.175.1.3,1
r_table=10.8.0.0,16,130.175.1.3,1
#Route to Tap Board
r_table=0.0.0.0,16,130.175.1.5,1
  Parms = {
    name= Board 4 Outside Router 0
  }
  /parms
/router
/board
#####Board 5#####
board=5
connections={
  R0,0 => O
  R0,1 => B0, G, IP
}
/connection
device=router,0
if_out=0,10.0.0.0,16,00:00:0c:05:00:00
if=1,130.175.1.5,24
  ##### R Tables #####
  r_table=10.0.0.0,16,10.0.255.254,0

```

#Route to Internal Boards

```

r_table=33.96.5.0,24,130.175.1.1,1
r_table=200.2.96.0,24,130.175.1.1,1
r_table=79.5.64.0,24,130.175.1.1,1
r_table=104.190.101.0,24,130.175.1.1,1
r_table=6.87.159.0,24,130.175.1.1,1
r_table=73.19.46.0,24,130.175.1.1,1
r_table=89.65.43.0,24,130.175.1.1,1
r_table=168.84.5.0,24,130.175.1.1,1
r_table=175.86.94.0,24,130.175.1.1,1
r_table=128.45.126.0,24,130.175.1.1,1
r_table=96.2.101.0,24,130.175.1.1,1
r_table=2.104.163.0,24,130.175.1.1,1
r_table=201.53.2.0,24,130.175.1.1,1
r_table=61.14.60.0,24,130.175.1.1,1
r_table=198.45.10.0,24,130.175.1.1,1
r_table=64.91.37.0,24,130.175.1.1,1
r_table=13.46.179.0,24,130.175.1.1,1
r_table=164.85.26.0,24,130.175.1.1,1
r_table=185.50.64.0,24,130.175.1.1,1
r_table=178.231.85.0,24,130.175.1.1,1
r_table=64.39.3.0,24,130.175.1.2,1
r_table=201.203.200.0,24,130.175.1.2,1
r_table=64.5.53.0,24,130.175.1.2,1
r_table=49.49.33.0,24,130.175.1.2,1
r_table=33.96.50.0,24,130.175.1.2,1
r_table=76.5.61.0,24,130.175.1.2,1
r_table=45.65.85.0,24,130.175.1.2,1
r_table=159.75.35.0,24,130.175.1.2,1
r_table=178.98.45.0,24,130.175.1.2,1
r_table=197.45.10.0,24,130.175.1.2,1
r_table=5.0.80.0,24,130.175.1.2,1
r_table=190.100.60.0,24,130.175.1.2,1
r_table=108.64.203.0,24,130.175.1.2,1
r_table=203.96.251.0,24,130.175.1.2,1
r_table=52.135.80.0,24,130.175.1.2,1
r_table=82.46.91.0,24,130.175.1.2,1
r_table=123.65.47.0,24,130.175.1.2,1
r_table=201.34.64.0,24,130.175.1.2,1
r_table=168.82.19.0,24,130.175.1.2,1
r_table=134.86.70.0,24,130.175.1.2,1
r_table=49.10.0.0,16,130.175.1.3,1
r_table=140.183.234.0,24,130.175.1.3,1
r_table=217.30.40.0,24,130.175.1.3,1
r_table=148.48.0.0,16,130.175.1.3,1
r_table=68.32.0.0,16,130.175.1.3,1

```

```
r_table=12.110.0.0,16,130.175.1.3,1
r_table=10.8.0.0,16,130.175.1.3,1
#Route to Proxy Board
r_table=0.0.0.0,16,130.175.1.4,1
#Route to Tap Board
r_table=0.0.0.0,16,130.175.1.5,1
Parms = {
    name= Board 5 Outside Router 0
}
/parms
/router
/board
```