Iowa State University
Digital Repository

Retrospective Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2007

Techniques in placing network monitors

Yongping Tang
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Computer Sciences Commons

Recommended Citation
Tang, Yongping, "Techniques in placing network monitors" (2007). *Retrospective Theses and Dissertations*. 15573.
https://lib.dr.iastate.edu/rtd/15573

This Dissertation 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 Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

المنارة للاستشارات

www.manaraa.com

Techniques in placing network monitors

by

Yongping Tang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Program of Study Committee:
Thomas E. Daniels, Major Professor
Dianne Cook
Yong Guan
Douglas W. Jacobson
Steve F. Russell

Iowa State University

Ames, Iowa

2007

Copyright © Yongping Tang, 2007. All rights reserved.

www.manaraa.com

UMI Number: 3289375

UMI®

UMI Microform 3289375

Copyright 2008 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

المنارة للاستشارات

www.manaraa.com

TABLE OF CONTENTS

المنارة للاستشارات

www.manaraa.com

المنارة للاستشارات

www.manaraa.com

المنارة للاستشارات

www.manaraa.com

LIST OF TABLES

المنارة للاستشارات

www.manaraa.com

LIST OF FIGURES

للاستشارات المنارة

www.manaraa.com

المنارة للاستشارات

www.manaraa.com

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis.

First and foremost, I would give me great thanks to my Major Professor Dr. Daniels. For these years, his guidance, patience and support inspired me to continue my research. His insights and great ideas lead the way of my progress.

Next, I would like to thank Dr. Guan for his great support of my research work. I would also like to thank my committee members for their efforts and contributions to this work: Dr. Jacobson, Dr. Russell and Dr. Cook.

Last but not least, I would like to dedicate this thesis to my wife Lei Zhu, without her support I would not have been able to complete this work.

المنارة للاستشارات

www.manaraa.com

ABSTRACT

Network monitoring systems are important components to protect networks. Due to large amount of data captured by monitors and the economical and technical constraints in adding monitors into network, research in monitor placement techniques is critical in network security.

In this thesis, we first present a network monitor structure with the ability of data reduction to reduce the space requirement of monitor systems. Then we discuss the techniques for the optimal placement of passive monitors in AS level and router level topologies where there are constraints on the number of available monitors for deployment. Arithmetic coding is used in data reduction and statistical models of the data being captured are provided. For monitor placement in the AS level topology, we first define average entropy and worst-case entropy to describe the remaining uncertainty of locating the origin of an attack given the monitors work perfectly, and use edge observed graph to present the possible deployment. Heuristic methods based on graph centrality to find the optimal placement with the minimized entropy is provided. For monitor placement in the router level topology, we set up a network model including routing strategies and threat model for generic network topologies and define the monitor placement problem as maximizing the observation of attack events. Results of heuristic solutions and greedy algorithm are compared. Monitor placement problems for both levels of topologies are proved to be NP-complete problems.

Because of the existence of asymmetric routing in Internet and the need of bidirectional data for stepping-stone analysis, we extended our research to capture bidirectional traffic via monitor placement. Greedy placement algorithms are provided for bi-directional traffic capturing. Experiment results are compared to demonstrate the tradeoffs in these algorithms. In our research, we also notice that it is hard to get network topologies with routing information,

www.manaraa.com

especially asymmetric routing information. As a result, we introduce asymmetric routing simulation techniques, which use the network topology generator as the data source, and provides simulation algorithms to generate the asymmetric routing on these topologies.

The contribution from this research include: (1) network monitor with data reduction, (2) monitor placement in AS-level network topologies, (3) monitor placement in router-level network topologies, (4) algorithms to capture bi-directional traffic through monitor placement and (5) simulation of network routing asymmetry.

المنارة للاستشارات
www.manaraa.com

CHAPTER 1. OVERVIEW

Networks have become ubiquitous and part of the critical infrastructure in today's world. More and more people rely on network for their work and study. Huge volumes of important information such as personal identity is transferred on the Internet all the time. While at the same time, with the development of the network, the number of threats and attacks also rises rapidly. These threats and attacks demonstrate that networks are vulnerable to attacks and misuse. Every year, companies lose billions of dollars due to network security related crimes. As a result, the safety of network has become a critical issue for everybody who uses the network.

Currently, a major threat on the Internet is the use of stepping stones and other types of proxies for covert illegal access to computers. In those attacks, attackers use a chain of machines as stepping stones before they attack their ultimate victims, so that they can hide their origins (show as Figure 1.1). To identify the complete attack path and trace the attack origins, stepping stone analysis systems are necessary. In these systems, traffic statistics and timing are recorded at many nodes in the network, correlation analysis is applied to link traffic observations with network attack sessions.

An obvious precondition to this analysis is the sufficient attack information recorded through some network monitoring systems. For example, most stepping stone research needs to monitor each session in at least one direction between each node in the attacking chain. Complicating matters, a new method of stepping stone analysis [27] has dramatically improved accuracy by requiring bi-directional traffic be monitored.

For network monitoring system, two types of questions need to be answered: the first one is how the monitoring system works, which includes what type of information the monitors

www.manaraa.com

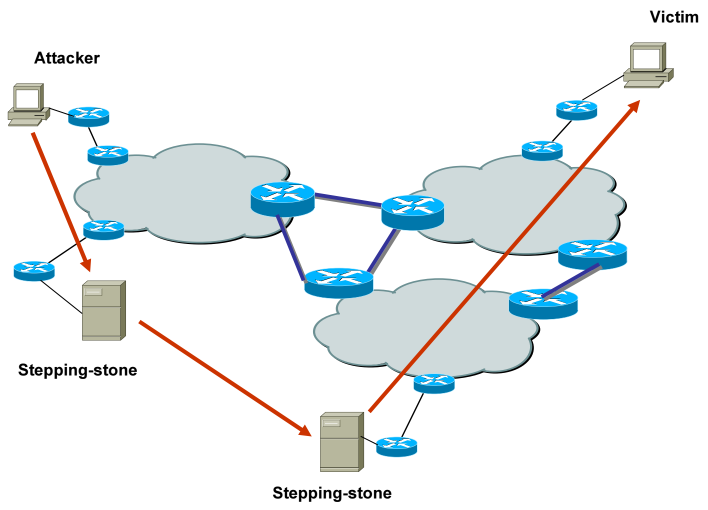


Figure 1.1 Stepping-stone Attack

www.manaraa.com

capture, what techniques the monitors use to capture the information, and how to management the storage of the huge volume of information being captured; the second one is how to deploy the monitoring system into the network because of the significant cost these monitors add into networks, economic tradeoffs among the number of monitors, the deployment mechanisms, and the systems effectiveness need to be considered.

In this research, we will discuss both of these two aspects. We will first provide a simple structure of the network monitor and the data reduction techniques; then we will discuss the problem definitions and solutions on how to place monitors in AS-level and router-level network topologies, and the algorithms to capture the bi-directional traffic via monitor placement; motivated by lacking of data-set in network research work like ours, we also discuss how to simulate the network asymmetric routing for both intra-domain and inter-domain routing. The contributions from this research include:

- Network monitor with data reduction
- Monitor placement in AS-level network topologies
- Monitor placement in router-level network topologies
- Algorithms to capture bi-directional traffic via monitor placement
- Simulation of network asymmetric routing

1.1 Research Motivation

During the past few years, we are working on an project called "Advanced Attack Attribution", which covers three areas: Stepping-stone attack analysis (SSA), attack attribution based on evidence graph (AAE) and network monitor related research problems. As we mentioned before, the pre-condition for both SSA and AAE is the sufficient network traffic data (TCP level or even network level packets) recorded via a network monitoring system. Therefore, the network monitor related research becomes the reseach assumption of SSA and AAE.

There are many open topics in monitoring systems, the most basic questions are how the monitor capture data and manage the storage of the huge volume of data being captured. To

www.manaraa.com

solve these problems, in this dissertation we present a simple monitor structure and the data reduction techniques applied to it.

Because of the cost the monitoring system added to network, a proper strategy of monitor placement is necessary. Based on different network abstraction, we discuss the techniques of placing monitor in AS-level and router-level network topologies. Our SSA research presents a high accuracy method, [27] which assumes that bi-directional traffic is recorded. Our monitor placement research is extended to provide placement algorithms to capture the bi-directional traffic data when there is routing asymmetry.

Many network research assumes network topology and routing information are available. However, in the real world, this information is difficult to obtain and even harder to be tailored for individual research, especially when asymmetric routing information is needed. We noticed this problem when we were doing the bi-directional traffic capturing through monitor placement research. This leads us to start the asymmetric routing simulation research.

1.2 Organization of the Dissertation

The rest of the dissertation is organized as follows:

Chapter 2 provides a simple structure of a network monitor, which has the function of data reduction to save the storage space. The technique we use in data reduction is network traffic compression using arithmetic coding. Since one network monitor can only oversee a limited scope of network, attack attribution need all monitors to work coorporately. In this chapter, we also present a simple architecture under which attacking events occurred at different locations can be associated together.

Chapter 3 presents the techniques to place monitor in AS-level network topologies. For a given number of monitors and a specific network topology, average entropy and "worst-case" entropy illustrate the remaining uncertainty in the origin of an attack when monitors work perfectly. One goal of this research is to optimize these two entropies. A brief proof that the worst-case deployment problem is NP-complete is presented. Moreover, greedy algorithms based on graph centrality heuristics to find high quality deployments are introduced. An

المنارة للاستشارات

www.manaraa.com

automatic monitor placement tool that implements these algorithms is developed and real network topology is used in the experiments to evaluate the results.

Chapter 4 introduces the mechanisms of placing monitors in a given router level topology to maximize the observations of attacking events. We set up a network model including routing strategies and a threat model in the generic network topology and define the monitor placement problem based on these settings. We give a simple proof to demonstrate that our problem is NP-complete and provide heuristic solutions with experimental results. Because of asymmetric routing, we extend our research to capture bi-directional traffic via monitor placement. We provide two greedy algorithms for bi-directional traffic capturing and compare the experiment results to illustrate the tradeoffs in these two algorithms.

Chapter 5 discusses the difficulties in asymmetric routing simulation. Two types of ”Early Exit” policy simulations are presented. The experiment results are compared with some previous research work.

At the end of each chapter, conclusions and and future works for the corresponding research topic are presented.

المنارة للاستشارات

www.manaraa.com

CHAPTER 2. A SIMPLE FRAMEWORK OF NETWORK MONITORING SYSTEM

Network monitoring system is the pre-condition of attack attribution. For most attack attribution analysis, we need network traffic information such as timing, packet headers or even payload to be captured by network monitors. A major difficulty for network monitor is the large storage space requirement for the captured information due to the huge amount of traffic on Internet. In fact, the space requirement is a problem for almost all log system. In our research, we want to provide a simple structure of the network monitor which will have the functionality of data reduction to save the storage space. The techniques to do the data reduction we will discuss is compressing the network traffic via arithmetic coding. Because one network monitor can only watch a limited scope of network, attack attribution need all monitors work corporately. In this chapter, we also present a simple architecture under which the whole system can correlate the events occurring distributed.

2.1 Introduction

Networks have become ubiquitous and part of the critical infrastructure in today's world. More and more people rely on network for their work and study. Huge volumes of important information such as finance information, electronic money and digital signature are transferred through Internet all the time. But at the same time, with the development of the network, threats and attacks also rose rapidly. These threats show that networks are vulnerable to attacks and misuse. Every year, companies lose a great amount of money due to network crimes. So safety of network has become a critical problem for everyone who uses the network.

To secure the network, generally, we have two aspects of network protection approaches.

المنارة للاستشارات
www.manaraa.com

The first is using defensive mechanisms to prevent the network criminals. This type of approaches typically needs to find out the network vulnerabilities and then to make patches to block these vulnerabilities, or to apply some special facilities to block malicious communication from outside. One example of these facilities that is most commonly used is the firewall. But this defensive method cannot completely solve the network safety problems. First, not only are we unable to find out all vulnerabilities, but also new vulnerabilities will always occur even when we patch old ones. In fact, the vulnerabilities in design and implementation of network are impossible completely avoid. With the rapid development of network technology, the structure of the network becomes more and more complex, thus giving greater opportunity for introducing vulnerabilities . The increasing complexity of systems also makes vulnerabilities much harder to find. Second, firewalls have limitations on the amount of state available as well as their knowledge of the hosts receiving the content and therefore cannot deal with the entire complex network environment.

As a result, other types of network protection approaches become more important. These approaches are not to block the network crimes but to find them, and to collect enough evidence of these crimes. Network criminals will be punished for their illegal actions thereby providing a deterrent to online crime. These methods are called network forensics. The main procedure of network forensics is to find out the the process of how the criminals execute their malice actions and to collect evidence to support an investigation. Such a procedure is called attack attribution. An effective network forensic system may increase the cost of the network crimes for the attacker and thus reduce the network crime rates.

Many criminals are never been found despite causing great losses. Even when they are found, attackers may waste investigators time and resources in manual examination of large log files. Without being punished for the lack of effective network forensics systems, criminals are un-apprehended. In fact, current network forensic investigations are often executed manually by examining different types of logs and using these logs to reconstruct the events that led to an attack. But due to the large volume of data in todays network, such manual methods make the analysis infeasible. We need infrastructures for forensic data collection, storage

المنارة للاستشارات

www.manaraa.com

and analysis. Although there are many logging mechanisms and audit trails kept on hosts, these event logs only care about what happens on the local machines without considering the relationship between local hosts and other network nodes. We need a network system to share the forensic information and do the correlation analysis in an integrated approach. The current network forensic model is also an after the process forensics procedure. Only after a criminal event happens, can such a model make a corresponding response through a long time and with human interventions. Further such a model usually cannot find the crucial evidence for network forensics because in the network environment digital evidence disappears quickly. To solve all these problems, it is necessary to provide a distributed system that can monitor large networks. This distributed system can collect and store the forensic data simultaneously at different locations, and exchange these data in a properly managed way to support the forensic analysis. And because the heavy data flow on the network, it is infeasible to store all the raw data of the traffic. So a data preprocessor is needed, which would use a proper data structure to store the forensic data and disseminate these data in a proper quantity. The manual network forensics model inevitably leads to long response time and missing the critical evidence. A practical network forensics system should be able to collect data in real time and use an automatic communication and coordination mechanism to provide a much quicker response to the adversary network events than possible manually.

In this chapter, we will give a framework of a network monitoring system. This system aims at providing a proper method to collect, store and analyze network events. It will also provide automatic evidence collection and quick response to network criminals . To find out the attackers hacking path, this system will support attack attribution by integrating some known attribution methods and provide an attack attribution graph generation mechanism to illustrate the attack. In general, our work aims at providing three contributions to network forensics: (1) An efficient data storage method based on distributed storage, pre-selection and specific data compression; (2) A general framework using distributed techniques supports easy integration of known attribution methods and effective cooperation within the system; (3) An attack attribution graph generation mechanism to illustrate hacking procedures.

www.manaraa.com

2.2 Related Work

Network monitors for data capturing share the same basic idea, they all work as the packet sniffer but may have different extended functionalities. To do packet capture, all these monitors are based on pcap [1] library which provides a high level interface to packet capture systems. All packets on the network, even those destined for other hosts, are accessible through this mechanism.

For log compression, Kimmo Hatonen et al. [2] provide a comprehensive log compression (CLC) method that uses frequent patterns and their condensed representations to identify repetitive information from large log files generated by communications networks. Their work formalized the log data and showed how the identified information can be used to separate and filter out frequently occurring events that hide other, unique or only a few times occurring events.

For network traffic capture, we know the data have specific format and we also wish one-time scan as mentioned above, so we use arithmetic coding [3] as the compressing algorithm. Arithmetic coding is a data compression technique originated from Shannon techniques. And it can get a best compression rate in theory. The basic idea of arithmetic coding is to assign a range to each symbol based on its probability within a probability line [0, 1]. It completely bypasses the idea of replacing an input symbol with a specific code. Instead, it takes a stream of input symbols and replaces it with a single floating-point output number. The longer (and more complex) the message, the more bits are needed in the output number. Due to the high cost of floating-point operation, all implementations of arithmetic coding use integer arithmetic to achieve a faster speed. To do the arithmetic coding, the data need a statistic model.

For network monitoring systems, there is one called SPIE [5] which uses Bloom Filter as the storage mechanism. SPIE is for IP traceback purpose so it only holds very little information. It is not fit for network forensics which need detailed network events recorded. Another system called ForNet [4] is more related to our framework. But it lacks of cooperative attack attribution and is more like a network data collection and query system.

المنارة للاستشارات

www.manaraa.com

2.3 System Description

In this section we give a detailed description of the system architecture of our distributed network monitoring system. This system will be deployed over a large network (such as a campus network) and will store the data in a distributed way but do the data analysis in a cooperating method.

2.3.1 Agent And Proxy Based Distributed System

In our framework, the monitors that collect data are called agents and there are also upper level monitors that manage the agents called proxies. Proxies are also data processing center and the place where the attack attribution will be executed. We deploy agents among the network to collect data and the data will be stored locally on the agents. Proxies get pre-processed data from agents and do the attribution analysis.

Each agent will monitor a small scope of network and we call such a subnetwork as an agent sub-net. Several agents will connect to a proxy that controls these agents. All the proxies are interconnected with each other, and if possible, a proxy should be deployed on an exclusive host that will just communicate with agents and other proxies, not do any other communications. Through this approach, the data traffic between one agent and one proxy and the data traffic between one proxy and another proxy will not affect the normal traffic of the network. Because one proxy will control several agents, it will then control several agent sub-nets. We call the collection of these agent sub-nets as a proxy sub-net. An agent only sends data to its control proxy. The connection between agent and proxy uses direct socket connection.

Proxies do not directly collect data from network. They just receive preprocessed data from agents. To avoid single point failure, we do not use a central control server but rather we treat proxies as peers. A proxy gets processed data will do data analysis and cooperate with other proxies. The communication between proxies is based on a distributed environment instead of socket because the cooperation between proxies is more complex than just static connections. If all the proxies can work together as a whole system, many correlation methods can be used to a larger scope and get more precise results. Figure 2.1 the basic illustration of

المنارة للاستشارات
www.manaraa.com

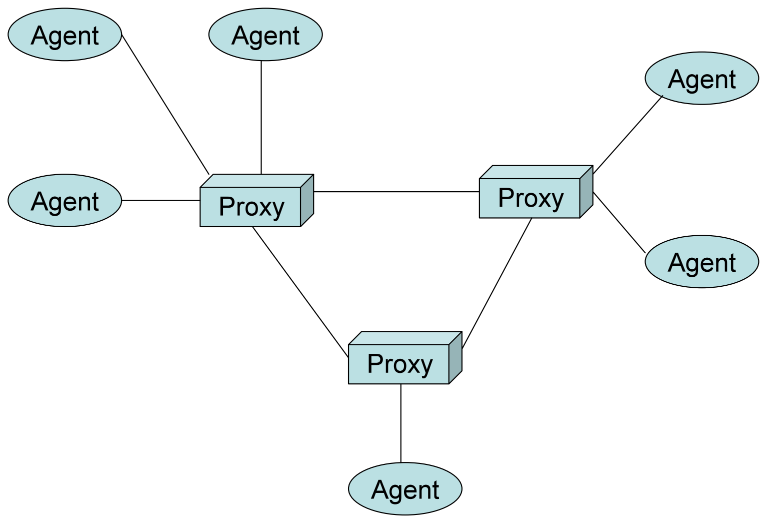


Figure 2.1 Architecture of the Framework

the system architecture. Using this architecture, we can store the data in a distributed way and cooperatively process these data.

2.3.2 Agent

Agents are responsible for data collection and simple analysis. They will be deployed on selected routers and monitor an agent sub-net. In our system, an agent has four functional parts: data collection module, data reduction module, data pre-processing module, and communication and control module (show as Figure 2.2).

المنارة للاستشارات
www.manaraa.com

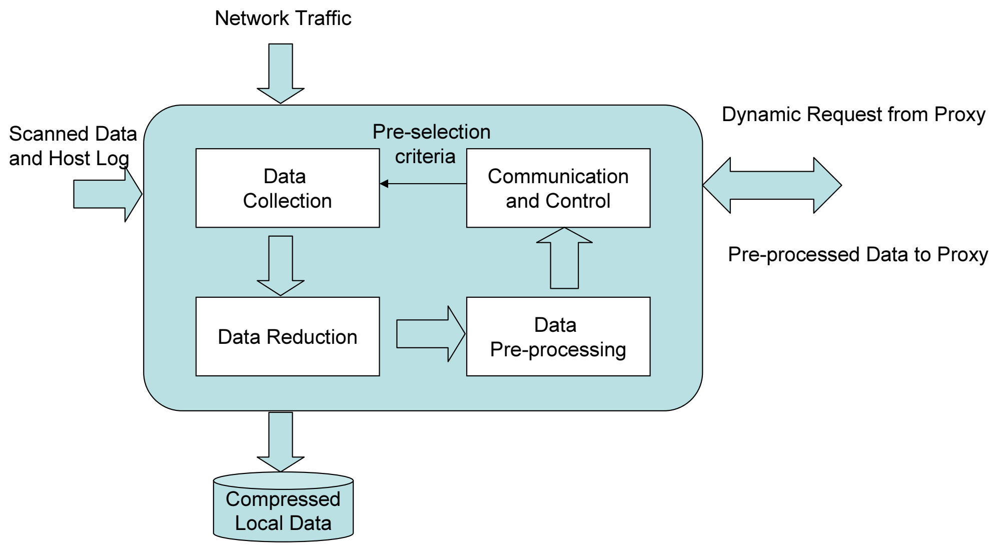


Figure 2.2 Agent Structure

www.manaraa.com

2.3.2.1 Data Collection Module

The data collection module is used to collect data from the network. Data collection will use both active and passive methods. The active method is to use a probe to periodically scan vulnerabilities based on known patterns and parse host logs.

The passive method is to listen on the network interface and capture the raw traffic data based on some pre-selection criteria. We know for most forensics problems only the packet header is needed for analysis. For example, the data for stepping stone analysis only need the timestamp of packets, basic connection information such as 4-tuple (source IP address, source TCP port, destination IP address and destination TCP port). Sometime, the sequence number, the acknowledge number and the flags in TCP header are also needed. So based on the pre-selection criteria, it only captures the needed part of a packet.

2.3.2.2 Data Reduction Module

The major difficulty for all network-logging systems is that the huge volume of the traffic data. It is impossible to store all the raw traffic data directly. In our system, we use three methods to deal with the data storage problem. The first method is the distributed data storage which is provided by the system architecture. Agents just store their local data and the data only relate to a small sub-net. Thus the data volume at one site will not be very large. The second method is the pre-selection criteria we mentioned above. It tells the agent only store the needed data. The third method we will address here is data compression using arithmetic coding.

We know most of the networks use TCP/IP over Ethernet and thus most of the network traffic packets are Ethernet packets. Network traffic data captured by libpcap library has a fixed format called TCPDUMP format, which consists of length indicators, packet timestamp and the Ethernet packet, which means that the data we captured has some specific structures. And because this data reduction module stores the data in compressed format to hard disk directly and continuously (may be as long as the work time of the system) , and there is no mid-step for compressing the existed data file, all the raw data can be read only once. So

المنارة للاستشارات
www.manaraa.com

the data here have two characteristics: specified data structure and one-time scan. We choose arithmetic coding to do the compression based on these two characteristics.

Arithmetic coding needs statistic model for data when doing the compression. We developed different models for different parts of the data and then use the basic coding engine to do the compression. Here we give some examples of the models we developed.

The first model is TimestampModel for the 8 bytes long timestamp. The first 4 bytes of the timestamp are seconds since Greenwich led the Western world out of the 1960s. The low 4 bytes are the absolute number of microseconds. Because 3 bytes are enough for the maximum value of the microseconds (999999 = 0x0f423f), the highest byte in these 4 bytes is 0. And it is clear that the arrival time of the packets is chronological, so the timestamp is in an increasing order. Thus the entropy of the timestamp is much less than 64-bit. We develop a model called TimestampModel for the timestamp. This model will encode the whole first timestamp (7bytes, the byte equals to 0 is omitted). From the second timestamp, this model computes the delta value from the previous timestamp. If the delta only use 4 bytes (1 byte for the difference of seconds and 3 bytes for the difference of microseconds), this model will adaptively encode these 4 bytes; otherwise, it falls back to encoding the 7 bytes long whole timestamp and begin a new loop repeated the above procedure.

There are two length indicators in the TCPDUMP packet, 4 bytes long each. For 10M Ethernet, only 2 bytes in the 4 bytes are used because the packets are not longer than 1514 bytes (not included the 4 bytes CRC checksum). The model for the length indicator that we develop is called LengthIndModel. It first omits the zero bytes. Based on our experiments, we found most packets are shorter than 128 bytes. The LengthIndModel model uses a cumulative history table with the alphabet 0 to 127. If the packet is longer than 127, the model takes 2 symbols for one indicator and a prefix symbol zero. For example, if the packet length is 56, we use the symbol 0x38. If the length is 1000, then 3 symbols will be used: 0x00, 0x03, and 0xE8. Although three one byte long symbols are used for two bytes long length, the probability of two bytes long lengths is much smaller than that of one byte long length, such an encode method is still efficient.

المنارة للاستشارات

www.manaraa.com

There are two MAC addresses in the Ethernet packet header. The first is the destination machine address and the second is the source machine address. We know that the MAC addresses in the Ethernet packet header will be changed at each machine. So if we capture packets at a certain machine, the MAC addresses we will meet are only the address of this machine and the addresses of the machines that directly connect to this machine. So the total number of the MAC addresses is limited and at most time, it is not very large. The MacModel use a dynamic allocated structure to save the MAC addresses. When it meets a new MAC address, a new node is allocated and the frequency of this MAC address is initialized to 1. If a MAC address already existed in the dynamic table, only the frequency will be increased 1. When do the encoding, a one-byte long symbol will replace the MAC address to be encoded. If the total number of MAC addresses in the dynamic table has exceeded 256 (in a large network), we need to refresh our model to ensure that we can still use one byte long symbol present the 6 bytes long MAC address. Through the above procedure, we can greatly reduce the entropy of the MAC addresses by first use much shorter symbols and second use arithmetic coding compression.

Other models for TCPDUMP data we developed are Ethernet type model, IP address model, ARP header model, IP header model, UDP header model, TCP header model and ICMP header model These models. Using arithmetic coding, it can achieve a much better compression ratio than the most commonly used compression tools such as gzip and the space needed for storage is much less than that the raw data need. Table 2.1 shows some of the results using arithmetic coding to compress the TCPDUMP format data.

Table 2.1 Compression Results by Arithmetic Coding

Part	Raw Size	gzip -9 Size	gzip -9 Ratio	AC Size	AC Ratio
Timestamp	15776784	4824212	30.58%	2924257	18.53%
Length-Indicator	7526400	495433	6.58%	309760	4.11%
Ether Type	3944196	4614	0.12%	2228	0.05%
MAC Address	11289600	178627	1.58%	125128	1.11%
IP Header	18804880	5543238	29.47%	3496764	18.59%
ARP Header	10656	167	1.57%	69	0.64%
TCP Header	560240	203796	36.38%	126926	22.65%

www.manaraa.com

2.3.2.3 Data Pre-processing Module

The major responsibilities of the data pre-processing module are data pre-processing and data preparation. Although we use data reduction techniques to reduce the volume of the saved data, it is still impossible for an agent to send all its data to its control proxy. Our goal is to reduce agent-proxy connections as much as possible. This module will use a lightweight IDS such as snort to filter and pre-process the traffic data. If there is an attack in the captured data, snort will generate many alerts. Most of the alerts are repeated and can be combined together. Here we use an algorithm called Leader-Follower Algorithm to combine the raw alerts into hyper alerts. This algorithm is very simple, it based on a time window and read the raw alerts one by one and group them to some hyper alerts according to the information such as source address, destination address, start time, end time and attack class. After finishing data pre-processing, the agent will send the hyper alerts to the proxy to indicate there is an attack found. We call this procedure as the attack-driven message transfer. Also, sometimes the proxy will need more detail data from the agent. When the agent gets such requests from its control proxy, it will try to answer the query. For example, when a proxy wants to do a stepping stone analysis, it needs the necessary information from the agent. The agent then decompresses the saved data and extracts the data based on the TCP 4-tuple and time range. The output data will include TCP connection 4-tuple (source IP address, source TCP port, destination IP address and destination TCP port) and the packet timestamps within the connection. These data then send back to the proxy. We call such procedure as the query-driven message transfer.

2.3.2.4 Communication and Control Module

Communication and control module is the hinge between an agent and its control proxy. This module sends data to and receives data from the agents control proxy through socket. When it needs to send data, it will connect to a specific port on the proxy and then send the data. It also has a server socket.

When the proxy sends some requests to the agent, the proxy will connect to this server socket. When the agent receives data selection requests from the proxy, this module will

المنارة للاستشارات
www.manaraa.com

analyze the requests and convert them into pre-selection criteria and send the criteria to the data collection module. When the agent receives data query requests, this module will send the query requests to the data processing and analyzing module to do data preparation.

2.3.3 Proxy

Proxies are connected with each other and constitute an integrated platform for attack attribution. As we mentioned before, the proxies need to work cooperatively and the communications among them may be very complex, so we can implement the data analysis methods as some distributed services. Then all the proxies can work together seamlessly like a single node. Another strength of this distributed architecture is our framework is an open and a scalable environment. We can easily use our system on different size of networks just by using corresponding number of proxies and agents; we can easily add new attribution techniques in our framework by developing some new services.

2.3.3.1 Attack Attribution through Proxies

We define attack attribution graph $G = \{E, V\}$ for our system. Here V is the set of proxies that are related to the attack attribution, i.e. if one proxy is in V, it means something happened in this proxy sub-net is related to the attack. E is the set of some connections that belong to the attack. For example, if there is a telnet from proxy sub-net A to proxy sub-net B to do the illegal access, the attack attribution graph G then have two vertices A and B and a directed edge (A, B) with an attribution illegal telnet. The procedure to generate the attack attribution graph proceeds roughly as follows: when a proxy (the victim is in this proxy sub-net) finds an attack from a remote proxy sub-net (correlate the links by stepping-stone analysis), it will inform the remote proxy and share the related information. The remote proxy then checks the related information on its own historical data. Some special connections such as telnet, SSH and ftp will be concerned based on a time window because attackers often use these services. If suspicious events found, the remote proxy will contact other proxies if they are related to these events. And then such activities continue to other proxies. All the proxies

www.manaraa.com

will work simultaneously to generate a sub-graph of its own and the final attack attribution graph is then generated by combine all the sub-graphs. Although this graph may not give the final result of the attack attribution, it should give a lot of useful information about the attack that took place.

2.3.3.2 Sample Scenario

Here we use a sample scenario to illustrate how the proxies work cooperatively to generate an attack attribution graph. The scenario is executed in our lab's test bed and described as the following:

The test bed will have 2 sub-nets with 3 hosts each. Each sub-net will have one proxy and one agent. The agent will not capture the traffic data if the data are to or from proxy. We call one sub-net as sub-net A because it has proxy A and the other sub-net is B because it has proxy B. We make one host in sub-net B as the victim and it has a Windows DCOM buffer overflow vulnerability. We make one host in sub-net A as the stepping-stone. The attacker will first login into the stepping-stone through telnet. And then he will download a hacking tool from a public FTP site and use this hacking tool to gain illegal access to the victim.

The agent in sub-net B will find the attack because it will monitor the traffic in this sub-net. Then the agent will send the hyper alerts to proxy B based on the attack-driven message transferring. Proxy B will tell the attack information to proxy A in another subnet because the attack is from the sub-net managed by proxy A. When proxy A receives these attack related data, it will then check its data (by query its control agent) about some specific service. Here it will check the telnet connection, ftp connection and SSH connection information in its history data using on a time window. Then the proxy A will find there are a telnet and an ftp connection suspicious.

At the same time, proxy B will also inform the proxy A to do ON/OFF [6] analysis by sending it the stepping stone information. Proxy A will query the data from its agent based on the time information provided by proxy B. Proxy A will check all the input connection that have a similar time scope provided by proxy B and then get the all the related TCP

www.manaraa.com

connection information from its agent. Then it does ON/OFF analysis for all the connections it chooses compare to the series from proxy B. Then the proxy A will find the telnet form outside is a stepping stone connection. Proxy A then uses the suspicious ftp information and the stepping stone results as well as the attack connection to construct a graph. And proxy B will also construct graph to indicate the attack from proxy A. Proxy A will send its graph to proxy B and proxy B will combine all the graphs together in the end. The final graph will give the attack attribution path for the attack it finds. The process to construct the graph is in fact a distributed algorithm running on different proxies. Each proxy will use its local attack attribution result to construct a partial graph and finally all partial graphs will be merged together.

The process for this sample scenario is like the following: For proxy B, it only knows an attack from proxy A (in fact, from one host managed by proxy B to one host managed by proxy A). So it just constructs a graph with two vertexes: v presents proxy B and u presents proxy A. And a directed edge from u to v presents attack from u to v.

For proxy A, the situation is a little complex because it has two suspicious connections: a stepping stone connection, ftp connection. So it will construct a directed graph with four vertexes: u presents itself, v presents proxy B, x presents the stepping stone source and y presents the ftp server. Directed edge (u, v) presents it has an attack to proxy B; (x, u) presents the ingoing stepping stone connection and (u, y) presents a ftp get to a ftp server somewhere. The partial graphs created by proxy A and B are shown in Figure 2.3.

Then the two partial graphs merged at proxy B. And the final graph may not very correct but it gives the some attribution results.

2.4 Conclusions And Future Work

In this chapter, we first provided a data reduction method using arithmetic coding and it has a good performance compare to gzip. In fact, this method can apply to any log with special formats. We just need develop statistic models for the formats, the basic idea is same. We also present a distributed monitoring system which can do attack attribution in a cooperate

المنارة للاستشارات
www.manaraa.com

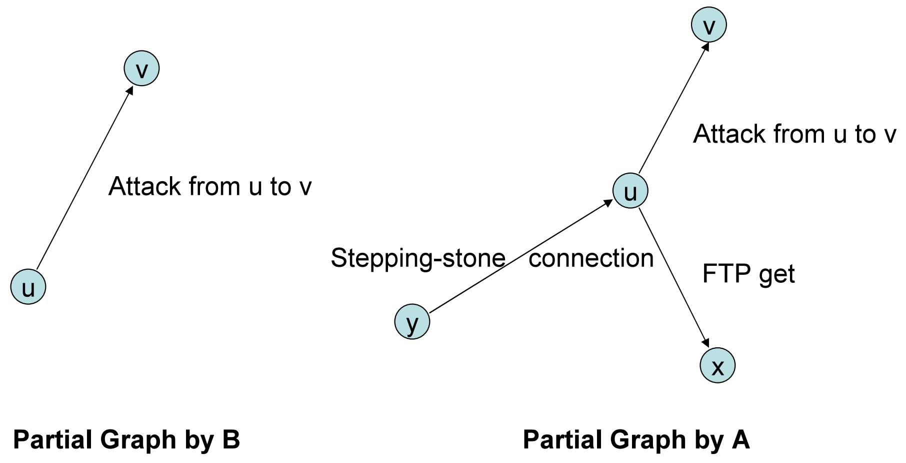


Figure 2.3 Partial Graphs Created by Proxies

www.manaraa.com

way. By merge the partial attack attribution graphs created on different proxies, we can find out the attack source and path automatically.

There is much remains to be done to for this framework. The first issue need to consider is the security of the system itself. If some agents or proxies are compromised, they will provide false data and even block the attack attribution. We need provide some mechanisms to identify compromised monitors before any communication. The second thing we will do in the future is to improve statistic models for TCP header in date reduction. Current TCP header model is a simple composite model which contains IP address model, sequence number model. Other parts in TCP header are treated as a single model based on repeated pattern. We need further provide models for small parts to gain a better compression ration.

المنارة للاستشارات

www.manaraa.com

CHAPTER 3. MONITOR PLACEMENT FOR ATTACK ATTRIBUTION IN AS LEVEL TOPOLOGIES

The precondition for attack attribution analysis is to record network events through network monitors. Little work has been done on how to place monitors in network. In this chapter, we propose the technique for the optimal placement of passive monitors in an AS level network where there are constraints on the number of available monitors for deployment. The placement problem is defined in terms of information theory metrics: for a given number of monitors and network topology, average entropy and "worst-case" entropy that describe the remaining uncertainty in the origin of an attack when monitors work perfectly are considered as the optimal object. A brief proof that the worst-case deployment problem is NP-complete is presented. Greedy algorithms based on graph centrality heuristics for finding high quality deployments are introduced to solve this problem. An automatic monitor placement tool which implements our approach is developed and we use real network topology in the experiments to evaluate our results.

3.1 Introduction

Network attack attribution is an important field in network security research. Currently, one major threat on Internet is illegal access. Although some attackers may hack into systems just for fun and without doing anything, others may make damage to the target system and steal critical information. To hide their origins, these attackers may access to intermediate hosts before hacking into the final system. These intermediate hosts are called stepping-stones. With system logs of the target host, only the nearest stepping-stone can be found. Stepping-stone analysis aims at determining whether two different links are correlated, and whether these two

المنارة للاستشارات

www.manaraa.com

links are part of the same stepping-stone attack. Therefore, to trace the origin of the network attack, stepping stone analysis is very important.

The precondition for stepping-stone analysis is that the information about these attack related links is recorded, which is often achieved by a network monitor system that has many monitors deployed in the network to collect network events. For example, [6] use Telnet trace to do the famous ON/OFF analysis. Our paper focuses on how to place monitors to provide support for stepping stone analysis.

In network attack attribution, Network Monitor System is deployed in the network to record information of network events to help identify the origin of network attacks. The more information is collected by the monitor system, the more accurate the attack attribution will be. However, due to efficiency reasons, it is impossible to deploy monitors on all network nodes. In most cases, the number of network monitors is much less than the number of network nodes. Therefore, a proper monitor placement strategy is important for the success of network attack attribution.

In this chapter, we define the problem of efficient passive monitors placement in the network so as to maximize the attained information about the location of the attack using a given number of monitors. To limit the number of monitors is important because it can reduce cost and computation overhead on the nodes, while at the same time maintaining the effectiveness of attack attribution. Determining where monitors should be placed depends on a number of criteria, which include the following:

- Topology of the network
- Number of nodes in the network
- Restrictions on the number of monitors
- The degree of each node
- Distance of each node from all other nodes in the network
- How many paths pass through one node

www.manaraa.com

- Traffic load forwarded by the node, etc.

Our work focuses on using information about the first six criteria mentioned above to develop a method to achieve an effective placement in any given network with a constraint on the number of monitors. Real life network maps are analyzed in our work with different monitor placement criteria, and the outcomes are compared to identify more desirable placements approaches. There two type of network topologies:

- AS Level Topology, where each node represents an autonomous system. (A node in the topology graph may also represent a provider point of presence that usually consists of several interconnected routers, in that case, we call it a POP level topology).
- Router Level Topology, where each node represents one of the provider's routers.

In this chapter, we treat each node equal, that means each node can be an attack source or stepping stone and traffic can be sent from any node and received at any node. So the AS level topology is more fit for our model. In next chapter, we will discuss how to place monitor in a router level topology.

A tool that implements the algorithm called "Topology Analyzer for Monitor Placement" (TAMP) is developed to help place monitors automatically for our work. TAMP is based on LEDA [48], a C++ class library for graph manipulation.

The rest of the chapter is organized as follows: in section two, some related work on placement problem is discussed, in section three and section four, the problem of optimal monitor placement is defined and the solution to the problem is presented. Then in section five, a simple proof is given to show that the optimal monitor placement problem is NP-complete and the complexity of the problem is discussed. In section six, some experiment results on real-life network are given. Finally, section seven draw some conclusions and propose some future works.

المنارة للاستشارات

www.manaraa.com

3.2 Related Work

Some works have been done on the placement problem. Most of these focus on replicated server placement or cache placement. Although these works are not related to network security, they have some general notion of coverage same as our work.

In general, server placement [28] is treated as the K-median problem, which means to choose k nodes in a n nodes network to serve other nodes acting as clients. The goal is to minimize the total weighted distance between all servers and all clients. [20] and [21] show such problems are NP-hard for general graphs.

A more complex situation is discussed in [10], which uses G=$\{V, E, H\}$ to represent a network with node set V, link set E, and H is a subset of V. The problem is interested in placing servers at some of the nodes in H, which will serve requests originated by any of the nodes in V. This work focuses on optimally selecting the locations where the servers must be placed, the proportion of traffic that must be routed by each client to each of the servers, and the routes that is followed by the requests issued by a client to a server. It uses round-trip delay, bandwidth consumption and packet routing as the QoS constraints, and decomposes the problem to three independent sub-problems.

In [11], another situation of server placement is discussed. It focuses on the case where both client and server densities are high. It considers the server placement can be regarded as a high-rate vector quantization problem with dimension two. The key idea of the model in work [11] is to regard the location of a request as a random variable with a probability density that is proportional to the demand at that location, and the problem of server placement as source coding, i.e., to optimally map a source value (request location) to a codeword (server location) to minimize distortion (network cost). It then gives a high-density model from single website to multiple websites. It defines $d(i, j)$ as the distance of serving a request of node i by a server located at node j, and turns the original problem into an optimization problem of calculating network cost defined as $C(K) = min_s \sum_{k=1}^{K} \sum_{j \in V} r(j)d(j, s_k)$.

These server placement problems are all optimization problems based on network topology that aim at minimizing the total network cost. The network cost in fact is weighted distances

المنارة للاستشارات

www.manaraa.com

between servers and clients. The purpose of these works is to provide a better performance of network communication.

Our work is also an optimization problem based on a given network topology. But the purpose is not for network performance but for network security. We use entropy as the optimizing object and aim at finding out the set of locations for a given number of monitors so that these monitors can minimize the uncertainty of the origin of attack. Also, unlike the above server placement problems, our model uses information theory metrics, and it is difficult to get polynomial or pseudo polynomial solution. We use heuristic method to find the solution.

Currently, little work has been done on network monitor placement problem. Most of work about the network monitors focus on how the monitor works. For example, Source Path Isolation Engine (SPIE) [5] is a Hash-based IP Trace back tool that is used to trace the origin of individual IP packets. Also, a lot of research has been done on IDS such as Snort that is based on a packet sniffer. The following sections will set up the optimization model and present our method to solve the problem.

3.3 Problem Formulation

Before defining the problem, some assumptions need to be made about the monitors and the network to simplify the problem and make it clearer. These assumptions are:

- The network topology graphs are connected graphs.
- The monitors are passive.
- The monitors are internal so that all traffic on any edge incident to a monitored node can be observed.
- The monitors are trusted, i.e. we do not consider the situation that a monitor has been compromised.
- The monitors are correlated with each other, i.e. they can link their observations to those of all neighbor monitors.

المنارة للاستشارات

www.manaraa.com

- The monitors can recognize traffic that originates from the nodes where they are placed

3.3.1 Problem Model

In AS level topologies, each node presents several routers (POP) or even an autonomous system and each node can be traffic source and destination, so each node has a probability to be an attack source. Our work aims at locating the attack source by place monitors in the network.

We describe the monitor placement problem in AS level topologies as the follows using graph-theoretic notations: an undirected graph $G = \{V, E\}$ presents the network topology, where V is the set of nodes and E is the set of links that connect the nodes. Given the number of monitors k and $k < n$, we want to find out a sub-set of V that these k monitors can be placed to, and minimize the uncertainty of the network. We define this k nodes sub-set of V as *IN*. The network topology graph G then can be presented as $G = \{V, E, IN\}$.

To solve this problem, we introduce a simplification of the graph called the Edge Observed Graph (EOG) [7] which represents the network topology from the perspective of a set of monitors placed in the network. The EOG can be presented as $G' = \{V', E', IN'\}$, and is reduced from G so that every node in G' is either a monitor or adjacent to a monitor. Therefore, each edge in E' is connected to a monitor and that is why it is called edge observed graph. The following algorithm is used to compute the EOG if monitors have been placed in the network:

```
Input  : POP level topology $G(V, E, IN)$
Output: Edge observed graph $G'(V', E', IN')$
begin
    $V' \leftarrow IN$;
    $IN' \leftarrow IN$;
    Construct graph $G_{tmp} = \{V, E - \{(u, v) | u \in IN\}\}$;
    for connected component $\in G_{tmp}$ do
        $V' \leftarrow V' \cup \{v'\}$;
    end
    $E' \leftarrow \{(u, v) | u \in IN)\}$;
end
```

Algorithm 1: Algorithm to compute EOG

المنارة للاستشارات

www.manaraa.com

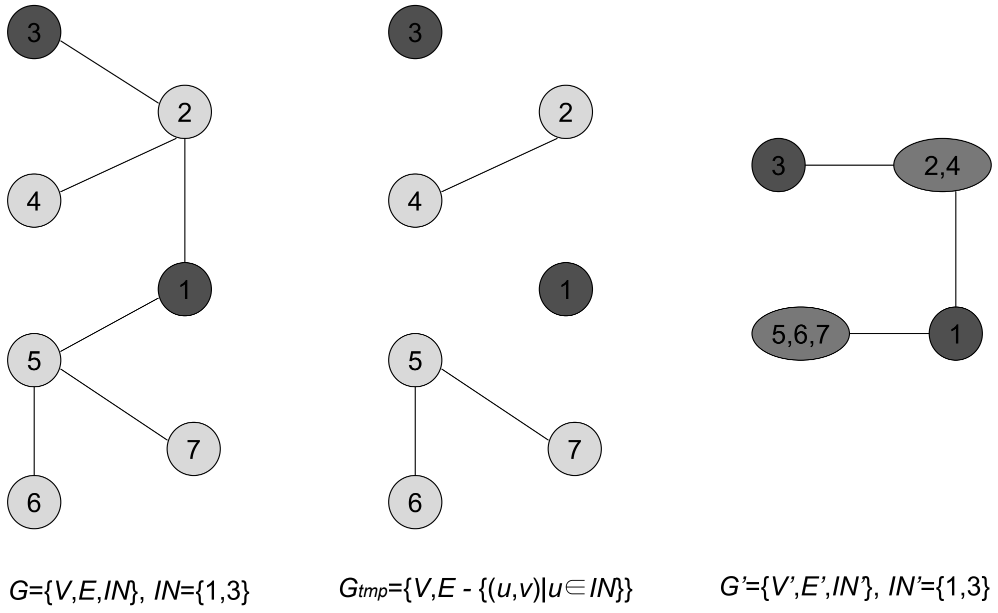


Figure 3.1 Construct Edge Observed Graph

www.manaraa.com

It should be noted that there is a correlation between traffic on different monitored edges, for example in Figure 3.1 if the monitor at node (3) detects attack traffic on the edge ((3) $\Longleftrightarrow$ (2,4)), and (1) detects attack traffic on edge ((1) $\Longleftrightarrow$ (2,4)) but not on ((1) $\Longleftrightarrow$ (5,6,7)), then the attack traffic must have originated from (2,4). However, if (1) detects attack traffic on both edges, then the attack could have originated from (2,4), and the same traffic passed through (5,6,7).

From the above algorithm, we know that in fact each EOG stands for a placement choice (if two placement choices make a same EOG, these two choices are equivalent). To compare the effects of different deployment choices, we use information theory metric entropy [18] as the standard of the choice. Here statistical entropy H is a probabilistic measure of uncertainty about which node the attack originated from.

Statistical entropy is a probabilistic measure of uncertainty. The entropy H is an expression of uncertainty about which node the attack originated from and is an indication of the number of bits that would be needed to ascertain the origin of the network attack. The higher the uncertainty, the more bits are needed. H = 0, if and only if the probability of a certain node being the origin of an attack is 1.

We consider the entropy of the network with regards to our knowledge of the origin of traffic. The entropy is dependent the factors which include the number of nodes (vertices), the number of components the network is broken into (edge observed graph), the number of monitors, and the probabilities of nodes being the origin of attack. Assuming any node in network topology graph G can be the origin of attack, there exists a probability distribution that for all node $v \in V$ has $p_o(v)$ which stands for the probability that v is the origin of the attack. It is clear that the smaller of the entropy of an EOG, the more information the monitoring system can provide to judge the attack origin (certainty). Therefore, the answer to our placement problem is to find the EOG with the smallest entropy, given the number of the monitors.

المنارة للاستشارات

www.manaraa.com

3.3.2 Metrics Definition

We define two types of entropies to measure the uncertainty: Average Entropy and Worst Case Entropy. Average Entropy means that the attacker knows nothing about the network and will not use information about the monitors to play his attack. Worst Case Entropy means that the attacker has knowledge of the network and the monitor placements. This attacker is considered to be a "smart" attacker who will always attack so as to maximize the uncertainty of the origin of the attack.

Because the node in EOG is either a monitor or a connected component in topology graph G, we call such a node as an analog and define analog = $\{V", E"\}$ where $V'' \subseteq V$. We use analog(v') to present the node set V" and for each node $u_j \in analog(v')$, we define $p_o(u_j)$ = probability of the node to be the attach origin in the original topology graph G and $p(u_j)$ = probability of the node to be the attach origin in the analog(v'). It is clear that $\sum_{u_j \in V} p_o(u_j) = 1$ and $\sum_{u_j \in analog(v')} p(u_j) = 1$.

Then we define the average entropy of the EOG as:

$$H_{AV}(G') = \sum_{v' \in V'} \sum_{u_j \in analog(v')} p_o(u_j) H(analog(v'))$$

if $p_o(u_j)$ is equal for all nodes in G, the average entropy becomes:

$$H_{AV}(G') = \sum_{v' \in V'} \frac{|analog(v')|}{|G|} H(analog(v'))$$

where $|analog(v')|$ is the number of nodes in the analog. The entropy of each analog(v') is given by $H(analog(v')) = -\sum_{u_j \in analog(v')} p(u_j) \cdot \log_2 p(u_j)$. If each node has equal probability to be the attack origin, we get $p(u_j) = \frac{1}{|analog(v')|}$ and $H(analog(v')) = \log_2 |analog(v')|$; if the nodes are not equally probable, $p(u_j) = \frac{p_o(u_j)}{\sum_{u_j \in analog(v')} p_o(u_j)}$.

For worst-case entropy, if all nodes are equally probable to be the attack origin, it can be calculated by using the entropy of the largest analog in the EOG, so $H_{WC}(G') = H(_{max}analog(v'))$. If the nodes are not equally probable, we define $H_{WC}(G') = max_{v' \in V'}(H(analog(v')))$.

We know an EOG means a placement choice and we also defined EOG's entropy that means how much the monitoring system can provide information to find the attack origin, so

www.manaraa.com

a feasible solution to our problem is to find an EOG with smallest entropy when compared to all other possible EOGs.

3.4 Problem Complexity

We will give a simple description to show that our monitor placement in AS level topologies problem is NP-complete using worst-case entropy criterion by reduce Graph Partitioning problem [19] to it.

We begin by noting that the placement of k monitors such that the resulting graph has worst-case entropy less than a bound p is in NP. A solution can be easily verified by finding the EOG and the analogs' corresponding entropies and ensuring they are all less than p. This can all be done in polynomial time so our problem belongs to the complexity class NP.

Assume we use worst-case entropy and all nodes are equally probable, we redefine our problem as the following: Given an undirected network graph $G(V, E)$, an integer k and a positive value m, we want to construct an EOG = G'(V',E',IN') that $H(analog(v')) < j$ for all $v' \in V'$. As we mentioned before, $H(analog(v')) = \log_2 |analog(v')|$, so if $|IN| < k$, we get $|analog(v')| < 2^j$. Since the nodes in the EOG are constructed by removing the monitored nodes and edges and combining the nodes in each subset to form a node in the EOG, we restrict ourselves to the number of edges that can exist in G' since all other edges will be removed when the nodes in each subset are combined, that is $|E'| \leq k$. We know that the Graph Partition problem is NP-Complete for a fixed $k \geq 3$.

So in this case, we are choosing locations for monitors such that the cardinality of the independent subsets created by removing the monitored edges is not larger than 2^j . Also, the only edges that can span different subsets are the monitored edges as all other edges are part of the independent subset and collapse into the nodes in the EOG. Since the Graph Partition problem is NP-Complete, even if all edge and node weights are 1, it can be seen that the weights of the edges in a sub graph is equal to the number of vertices in the sub graph and that the weight of the edges spanning the sub graphs is equal to the number of edges that span the sub graphs. At this point we can see that our problem with the assumptions is the same

المنارة للاستشارات

www.manaraa.com

as the graph partition problem. So we know our problem is also NP-complete.

3.5 Heuristic Solutions Based on Central Indices

To solve our NP-complete problem, we present several heuristic methods based on central indices.

The centrality of a node is a measure of its importance with respect to the rest of the nodes in the network. There are several centrality indices including degree, betweenness, closeness, eccentricity, radiality, status, eigenvector, pagerank, authority and hub [13]. The indices taken into consideration here are degree, betweenness, and closeness. Degree is chosen because it is a measure of the probability a node has of receiving information flowing through the network; closeness is chosen because it is a measure of how close a node is to all the other nodes in the network; and betweenness is chosen because it is a measure of the effect a node has on the movement of information between other nodes.

For a topology graph $G = (V, E, IN)$, the shortest distance between two nodes s and t is denoted by $D(s,t)$ and the number of shortest paths between two nodes s and t is denoted by $SP(s,t) = SP(t,s)$. More over, the number of shortest paths between two nodes s and t that pass through some other node v is denoted as $SP_v(s,t) = SP_v(t,s)$ [14]. Based on these notations, three centrality indices are defined as:

- Degree - This is a measure of the number of neighbors a node has. The degree of node v is denoted as $deg(v)$. The higher the degree of a node, the more information that is likely to pass through it and the greater influence it will have on other nodes in the network.

- Closeness - This is a measure of the total graph theoretic distance of the node to all other nodes in the network. The closer a node is to most of the other nodes, the faster it will receive information, a quality that is desirable for the placement of a monitor [12]. The closeness is defined as $C_C(v) = \frac{1}{\sum_{s \in V} D(s,v)}$.

- Betweenness - This is the fraction of geodesic paths between two nodes s and t that pass through the node v, where $s, t, v \in V$. A node with a high value of this measure

www.manaraa.com

will tend to have a lot of information pass through it. Betweenness is defined as $C_B = \sum_{s,t \in V, s \neq t \neq v} \frac{SP_v(s,t)}{SP(s,t)}$ [16].

In choosing the nodes where we want to place the monitors, our choice of nodes tends to be those with high values of degree and betweenness and low values of closeness. It is assumed that in these networks, any traffic going from node s to node t would travel along one of the shortest paths. The following is the recursive algorithm to find the placement choice:

```
Input  : POP level topology $G(V, E)$
Output: A monitor placement with k nodes
begin
    Compute centrality index(degree, closeness or betweenness) for nodes in $G$;
    for $k$ monitors do
        Choose a node based on index;
        Remove the chosen node and update $G$;
        Re-compute centrality index;
    end
    Compute EOG;
    Computer entropy for EOG;
end
```

Algorithm 2: Recursive Heuristic Algorithm to Find the Placement

If we assume the graph $G = (V, E)$ has n nodes and m edges, and k monitors are to be placed ($k < n$), we get the complexity for the heuristic methods as:

- For degree, we need to traverse all nodes to calculate the degree of all nodes, and during this traverse, each edge will be counted twice because the sum of degree equals twice of the number of the edges. Therefore, to calculate degree, the complexity is $O(n + 2m)$.
- The closeness is to find the shortest paths for all pairs of nodes. Therefore, if we use Floyd-Warshall algorithm, the complexity is $O(n^3)$.
- The betweenness is to run Bread First Search (BFS) from each node of the graph. Since the complexity of BFS is $O(n+m)$, the complexity of betweenness calculation is $O(n^2 + nm)$.

المنارة للاستشارات

www.manaraa.com

3.6 Experiment and Results

We use some real-life network graphs [17] as the experiment data set. Besides the recursive algorithms, we also apply non-recursive algorithms, that means directly choose k nodes by compute the index only once, to compare the results.

3.6.1 Final Entropies Based on Different Heuristics and Number of Monitors

For each of the network graphs, we calculated the metrics. We wanted to see whether we could partition each graph so that we had a number of partitions of at least 90% of the total number of nodes with the best result being a number of partitions equal to the number of nodes. With the number of partitions being equal to the number of nodes, we would be able to determine the origin of an attack with full certainty. We chose an acceptable benchmark of 90%, i.e. $0.9n \leq m \leq n$, here m is the number of partitions (analog).

We chose 3 or more monitors by taking a percentage of the total number of nodes from 10% to 50% and observe whether theses goals were achieved. The results are shown in Table 3.1, where H_C, H_B and H_D are the final entropies got based on indices closeness, betweenness and degree; m is the number of partitions, j_m is the maximum number of nodes in one partition and num_M stands for number of monitors.

Table 3.1 Entropies of real networks

H_C	m	j_m	H_B	m	j_m	H_D	m	j_m	num_M
	Colt Telekom (n=43); min(m)=38								
1.83	19	12	1.81	21	11	0.96	28	7	13(30%)
1.84	22	14	0.87	30	8	0.37	35	2	18(40%)
1.44	26	11	0.44	35	2	0.28	37	2	22(50%)
	Switch Network (n = 31); min(m)=27								
0.52	24	4	0.32	26	2	0.32	26	2	10(30%)
0.45	25	4	0.13	29	2	0.19	28	2	13(40%)
0.19	28	2	0	31	1	0.13	30	2	16(50%)
	Cable Wireless (n=33); min(m)=29								
2.61	11	12	2.78	10	15	2.76	12	20	7(20%)
2.02	15	11	1.81	17	13	1.50	18	10	10(30%)
1.46	19	9	1.11	22	9	0.53	26	4	14(40%)

www.manaraa.com

We were able to achieve our goal of $0.9n \leq m \leq n$ and $1 \leq j \leq 3$ with at most half of the nodes having monitors deployed on them, i.e. $m = n/2$ every time using the recursive method and every time except once using the non-recursive method. Generally, it was demonstrated that the percentage of times that the lowest entropy was obtained using the degree measure as compared to closeness and betweenness was relatively constant (70%) regardless of whether the nodes were chosen recursively or not.

3.6.2 Choose Better Entropy

We also did experiments to evaluate which method is better to get small entropy. Here H_{BEST} stands for the lowest entropy for each of the values of m from all the heuristics,

$$H_{BEST} = min(H_C, H_B, H_D, H_{REC\_C}, H_{REC\_B}, H_{REC\_D})$$

for each metric, and H_{AV_BEST} and H_{WC_BEST} represent the lowest entropy for the average entropy and worst case entropy metric respectively. The entropy for each of the heuristics was compared with that minimum value, and if for example, $H_{AV\_C} \leq H_{AV\_BEST}$, a weight w =1 would be assigned to average closeness for m = 3 and if $H_{AV\_C} > H_{AV\_BEST}$, w = 0 would be assigned to it. The weight values were added up, and the heuristic with the highest value of $\sum_m w$ was chosen as the best. Figure 3.2 show the results obtained.

We observed that recursive betweenness had the overall highest weight that indicates that it may be the best to use in placing our monitors.

3.6.3 Non-recursive v.s. Recursive

To further evaluate the performance of non-recursive methods and recursive methods, we compared their results to identify how much better one was than other. An example of some of the results obtained is shown in Figure 3.3.

We observed that on the average the recursive method yielded more favorable results than the non-recursive method but the differences varied from one heuristic to another. The percentage of times the lowest entropy was achieved using closeness and betweenness increased drastically when the monitor placements were chosen recursively.

المنارة للاستشارات
www.manaraa.com

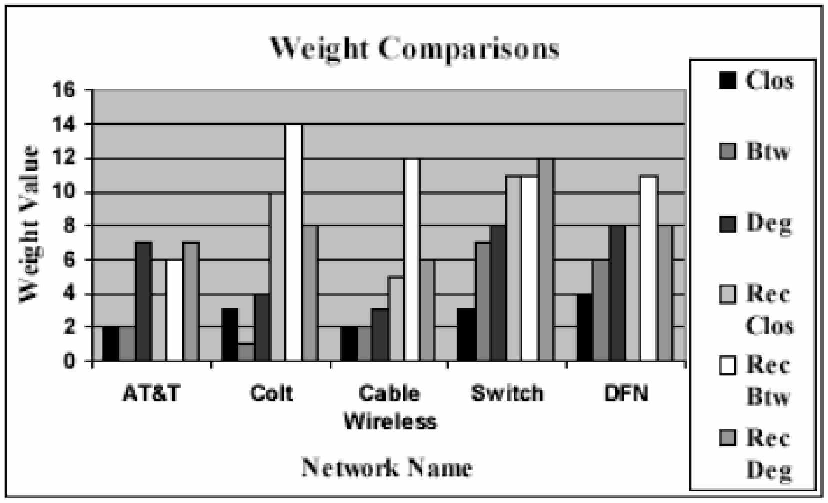


Figure 3.2 Comparisons of Weight Values

www.manaraa.com

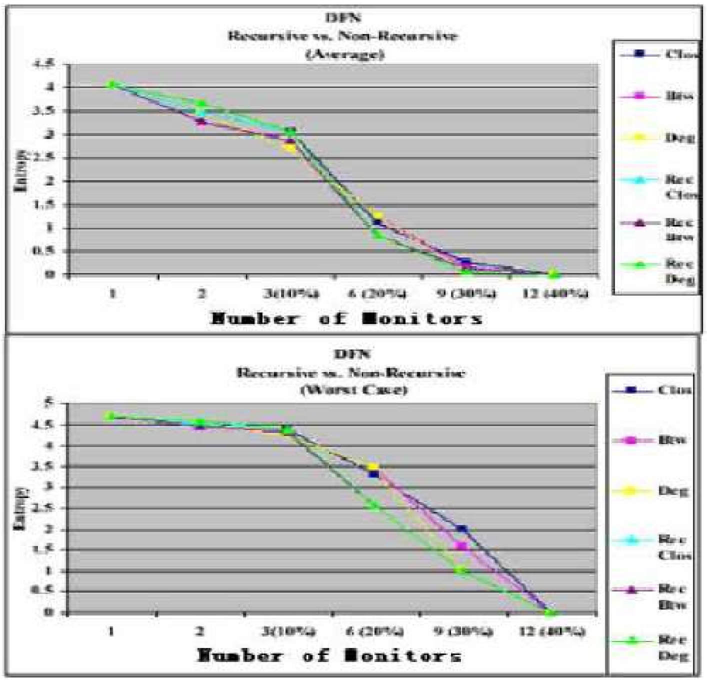


Figure 3.3 Comparisons of Recursive and Non-Recursive

www.manaraa.com

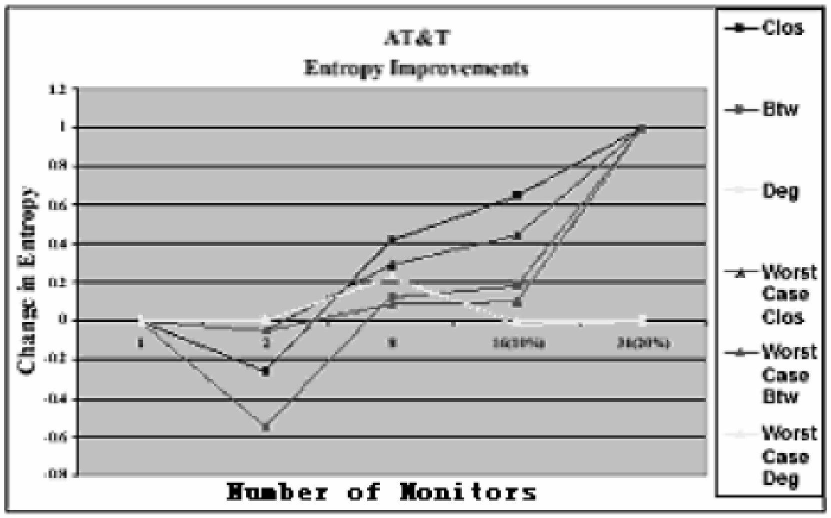


Figure 3.4 Comparisons of Entropy Improvements

We also generate graphs that compared the improvements (or deterioration) due to using the recursive method over the non-recursive method. This was done by subtracting the recursive entropy from the non-recursive entropy for each of the heuristics for the different values of m, e.g. $H_{AV\_C} - H_{REC\_AV\_C}$. The result is shown in Figure 3.4. We found on the average, the improvements in entropy for closeness and betweenness were more significant than that obtained for degree, which leads us to conclude that using the recursive method in applying the degree heuristic does not make a significant difference. We also observed that using the recursive approach improved the entropy for the average cases more than it did for the worst-case ones.

3.6.4 Conclusions

Based on the results above, we are able to conclude that in terms of the number of times we achieve the lowest entropy, degree is the best for the non-recursive approach alone and

www.manaraa.com

recursive betweenness is the best choice when consider the recursive method. And because the change in entropy for degree does not change significantly from the non-recursive approach to the recursive approach as compared to betweenness and closeness, recursive betweenness and recursive closeness are better choices than recursive degree.

3.7 Future Work

In this chapter, only the case where each node has an equally probability of being the origin of an attack is considered in the analyses of the real life network graphs. We are planning to analyze the case where monitors have unequal probabilities to see the effects of these metrics.

The work in this chapter is fit for AS (or POP) level topology because we treat all nodes can be attack origin. For router level topology, nodes are different and not all can be the attack source. We will provide model for monitor placement on router level topology in the next chapter.

المنارة للاستشارات

www.manaraa.com

CHAPTER 4. ON THE ECONOMIC PLACEMENT OF MONITORS IN ROUTER LEVEL NETWORK TOPOLOGIES

Due to economical and technical constraints, it is necessary to provide a proper monitor placement strategy. In the previous chapter, we discussed how to place monitors in AS level network topologies. In this chapter, we discuss how to place monitors for a given router level topology to maximize the observation of attack events. We set up a network model including routing strategies and a threat model for general network topology and then define the monitor placement problem. We give a simple proof to show that our problem is NP-complete and provide heuristic solutions with experimental results. Because of asymmetric routing, we also extended our problem to capture bi-directional traffic via monitor placement. We provided two greedy algorithms for bi-directional traffic capturing and compared the experiment results to show the tradeoff in these two algorithms.

4.1 Introduction

As we mentioned before, an obvious precondition to do attack attribution analysis is sufficient attack information recorded through some network monitoring system. For example, most stepping stone analysis research needs to monitor each session in at least one direction between each node in the chain. Complicating matters, a new method of stepping stone analysis [27] has dramatically improved accuracy by requiring bi-directional traffic be monitored. To get link level data traffic for these analysis, we need place monitors at router level.

Because monitors add significant cost to networks, there is the need to consider economic tradeoffs between the number of monitors, their placement, and the system's effectiveness.

Also, due to economic and technical constraints, it is impractical to do universal deploy-

المنارة للاستشارات

www.manaraa.com

ment. Furthermore, the power law degree distribution of networks suggest that there are highly effective, low cost placements. Finding these placements are important to making such stepping stone analysis and other types of attribution systems practical and cost effective.

Generally, there are many factors that affect how to place monitors. These include the network topology, routing strategies, threat models in the network, administrative or technological constraints for placement and the cost of deploying a monitor in a given location. Also, we can consider several different criteria to measure the effectiveness of the monitor placement. This can also drive the placement strategy. Figure 4.1 shows a general model for the monitor placement problem.

In this chapter, we take the amount of expected attack traffic observed as the metric of our monitor placement problem and define our problem so as to maximize the observation of attack events in a router level topology network model. The basic inputs for our problem are router level network topology and the number of monitors. In this simplistic model, we consider the number of monitors as the cost, but allow that router-specific characteristics (e.g. the amount of traffic passing through a node) may affect the cost of individual monitors. In this more general case, the sum of costs may be considered.

To take into account the diversified routing policy and complex attack situations in the Internet, we create a generalized routing and threat model. These allow us to study the problem without in depth information that an ISP or other agency deploying this work would have. It should be noted that given this information, it is straightforward to use it instead of our general assumptions. We set up a routing model based on OSPF [25] and provide a reasonable threat model. Our output is an optimized placement with the goal of maximized the observed attack events. We will show that this optimization problem is an NP-complete problem, provide some heuristic solutions, and compare experimental results.

In section six, we further extend this problem to consider how to observe bi-directional traffic. Due to the asymmetric routing in real networks, the data back and forth between a pair of nodes may not travel through the same path. We will provide a polynomial greedy algorithm for this issue and give some evaluations on the experiment results.

www.manaraa.com

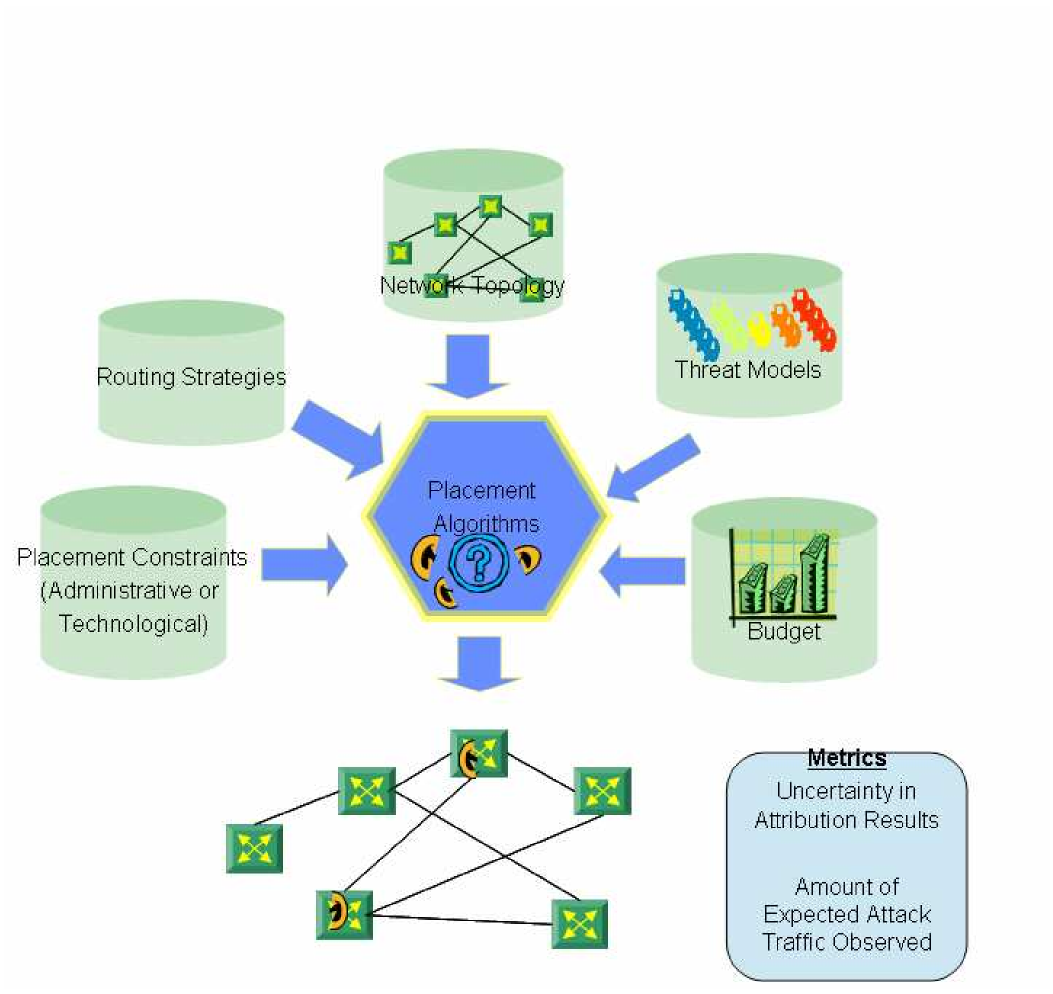

Network Topology
Routing Strategies
Threat Models
Placement Algorithms
Placement Constraints (Administrative or Technological)
Budget
Metrics
Uncertainty in Attribution Results
Amount of Expected Attack Traffic Observed

Figure 4.1 General Model for Monitor Placement

المنارة للاستشارات
www.manaraa.com

The rest of the chapter is organized as follows: in section two, we discuss some related work on placement problems; in section three, we set up a general routing model and a threat model and define the monitor placement problem based on these two models, and then a simple proof shows that our problem is NP-complete; in section four, we present heuristic solutions for our problem and compare experiment results.

In section five, we give a brief discussion of new algorithms to trace traffic through a monitored network, and in section six, we discuss how to place monitors to observe bi-directional traffic using a polynomial-time greedy algorithm. We then draw some conclusions and propose future work.

4.2 Related Work

For router level monitor placement work, there is some work about packet filter placement to defend against denial of service attacks [26]. The work assumes that the routing information is known for the network, and they want to place packet filters to drop all spoofed packets between any two nodes. The purpose of the packet filter placement is to minimize the number of filters and still ensure that any spoofed packets will be detected. It has proved the problem is a NP-complete problem by a reduction from Vertex Cover problem. The work also discussed some special cases (trees, self healing rings, and bipartite networks) that make the problem solvable in a polynomial time.

The network model it used is a directed graph $G(N, A)$, where N is the set of nodes in the network, $|N| \geq 3$, and A is the arc set. It defines the communication set C as the set of all node pairs that engage in the exchange of packets. A node pair $(u, v) \in C$ if and only if packets at $u \in N$ can travel to $v \in N$ under a given routing information R. A packer filter can drop source spoofed packets based on R. The problem is determines the minimum number of filters required to achieve perfect security, i.e., all packets with forged origin addresses are discarded based on the above given information (G and R).

The problem can be described using integer linear programming:

minimize $\sum_{i \in N} y_i$

www.manaraa.com

$s.t. \sum_{i \in N_{sod}} y_i \geq 1$

$\forall (o,d) \in C, \forall s \neq o : (s,d) \in C$

$y_i \in \{0,1\} \forall i \in N$

Another work about monitor placement is [49]. The difference of this work to our work is it try to place edge (the connection between two nodes) monitors instead of node monitors as we do. This work also treat IP network as an undirected graph $G(V,E)$. It defines the traffic flow (one-directional link) as packets source from same nodes, destination to same nodes, pass through same path. The purpose of this work is to find a subset of E to place monitors and also find a sample rate for each monitor.

The metrics used in this work are costs and reward:

- Deployment cost $C_D = \sum_{i \in L} f_i y_i$, where L is the set of all edges that allow to place monitors, it is clear $L \in E$; f_i is the monitor deployment cost at edge i; f_i is 1 if a monitor is placed at edge i.

- Operating cost $C_O = \sum_{i \in L} c_i y_i \sum_{j \in D} \rho_j m_{ij}$, where D is the set of all flows; c_i is the unit sampling cost at monitor i; ρ_j is the traffic demand of flow j; m_{ij} is the fraction of flow j sampled by monitor i.

- Monitoring reward $C_M = \sum_{j \in D} \mu_j(M_j)$, where M_j is the fraction of flow j sampled by monitors and $\mu_j(M_j)$ is the benefit gained by monitoring flow j, it is a non-decreasing and concave utility function.

By considering with or without sampling and different optimization goals, the problem can be further categorized into two types and four problems:

- Monitoring problems without sampling
  - BCMCP: Budget Constrained Maximum Coverage Problem
  - MDCP: Minimum Deployment Cost Problem
  - MDOCP: Minimum Deployment and Operating Cost Problem
- Monitoring problems with sampling

المنارة للاستشارات

www.manaraa.com

– BCMCP-S: Budget Constrained Maximum Coverage Problem w/ Sampling

All these problems can also be described in the format of integer linear programming and using existed algorithms to solve.

4.3 Problem Description and Complexity

We describe our monitor placement problem in the context of a given topology router level network. The monitor placement is to choose some locations (routers) where we can place monitors to record the network activities to fulfill some criteria.

For attack attribution, we need to record attack activities so that we can do correlation analysis later. Current illegal access attacks usually use several stepping-stones before finally hacking into the targets. The monitor system aims at recording as many attack activities as possible so that we can provide enough information to find these stepping-stones. We are not interested in the locations where we can monitor maximized general traffic; we are interested in the locations where the attack traffic will pass through with greater probability and monitoring it such that we cover as much distinct attack traffic as possible.

We cannot know exact attack traffic before it happens, but we can estimate the possible attack traffic distribution based on some factors such as the number of hosts behind routers, the history of attacks found on the routers and the location or functional features of the route and its hosts.

4.3.1 Threat Model

The monitor placement problem can be modeled using the following graph-theoretic approach. Let graph $G = (V,E)$ presents the network topology, where V is the set of nodes in topology and E is the set of links connecting these nodes. For routers in network, we know some of them are connected to both hosts and routers and the others are only connected to other routers. (Show as Figure 4.2)A router level topology is the presentation of these features. So in our model, we separate the node set V into two groups: the access node set D and the relay node set R. Access nodes are routers with access sub-nets, all hosts are located in these

www.manaraa.com

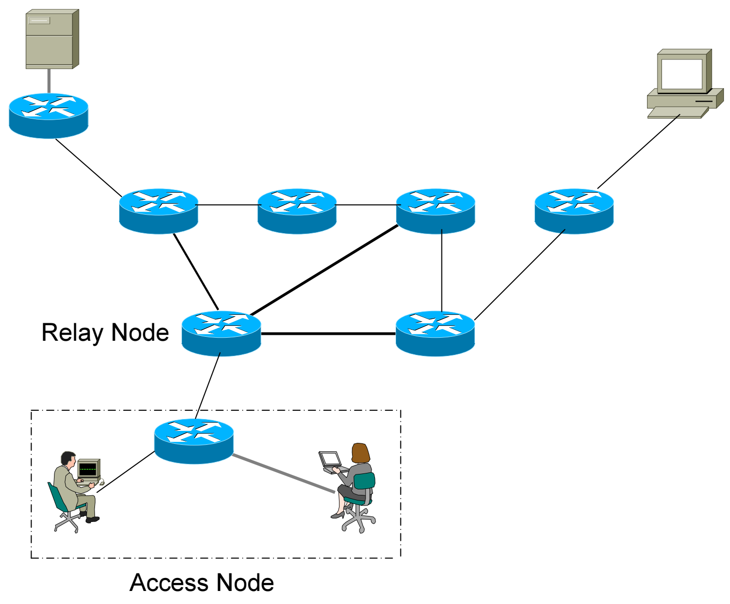


Figure 4.2 Network Model

sub-nets. So in the router level topology, we can deem all the traffic are source from access nodes and also destined to access nodes. Relay nodes are core routers that primarily transfer packets to other routers and do not have access sub-nets. Also, we assume relay nodes cannot be the attack source (stepping stone). It is clear that $D \cup R = V$ and $D \cap R = \emptyset$.

Because the monitor system aims to record potential attack traffic, from now we describe the network in the "attack only mode" i.e. we think of that all traffic chosen for capture are attacks. For an access node pair $(x,\ y)$, we define the attack rate $r(x,\ y)$ as the number of attacks from x to y in a unit of time. Also, for the access node x, it has an attack rate $r(x,$ *) that indicates the number of attacks sourced from x in the unit of time. It is clear that $r(x,*) = \sum_{y \in D, y \neq x} r(x,y)$. It is hard to know $r(x{,}y)$ because x may communicate with any y in the network and guessing the attack rate between them is akin to reading the source's mind!

www.manaraa.com

However, there are some clues to estimate $r(x,*)$, such as the number of hosts behind the access node x and the history of attacks related to the hosts in x's access sub-net. To simplify the simulation in the experiment of this paper, we assume $r(x,*)$ is known for any $x \in D$ and we assume the attacks are uniformly distributed in the network, which means $r(x,y) = \frac{r(x,*)}{|D|-1}$. Of course, if $r(x,y)$ could be measured or better approximated that would obviate this rough estimate.

For a relay node d, we define an attack observe rate $aor(d)$ as the number of attack events observed on by the node d in a unit of time. If detailed routing information is known, the attack observe rate can be decided based on it and $r(x,y)$. Unfortunately, routing information is hard to know and our problem only assumes the topology is given. So here we give a general approach to estimate the attack observe rate. We assume OSPF is the routing protocol and each node knows the topology. Then we can deduce that traffic between any access node pair will pass through the shortest path between the access node pair and if multiple shortest paths exist, the traffic will equally distribute among them. If $SP(x,y)$ is the number of shortest paths between access nodes x and y, $SP_d(x,y)$ is the number of shortest paths between access nodes x and y that pass through relay node d, we get:

$$aor(d) = \sum_{x \in D} \sum_{y \in D, y \neq x} \frac{SP_d(x,y)}{SP(x,y)} r(x,y)$$

As before, this is an estimate of routing behavior and more precise knowledge of the protocols or actual routing tables can be used to better model a given topology.

Here we provide Algorithm 3 to decide the attack observe rate for each relay node.

4.3.2 Problem Definition

Based on the above routing and threat model, we can define the monitor placement problem as the following: In the router level topology $G(D,R,E)$, find a k node subset of R such that the effective attack observe rate of the k monitored relay nodes is maximized.

Because the same attack event will sometimes be observed by multiple relay nodes along an attack path, we note that the sum of k nodes' attack observe rates is not the sum of k nodes' individual attack observe rates. We call this difference an overlap. For example, consider

المنارة للاستشارات
www.manaraa.com

```
Input  : Network topology $G(D, R, E)$, $r(x, *)$
Output: Each relay node's attack observe rate
begin
    for $x \in D$ do
        $aor(x) \leftarrow r(x, *)$;
    end
    for $d \in R$ do
        $aor(d) \leftarrow 0$;
    end
    for $x \in D$ do
        for $y \in D$ and $y \neq x$ do
            $aor(y) \leftarrow aor(y) + r(x, *)/(|D| - 1)$;
            $L \leftarrow$ the set of all shortest paths between $x$ and $y$;
            for $l \in L$ do
                for each relay node d on l do
                    $aor(d) \leftarrow aor(d) + \frac{r(x,*)}{(|D|-1)*|L|}$;
                end
            end
        end
    end
end
```

Algorithm 3: Attack Observe Rate for Relay Node

the case where two relay nodes are connected by an edge. If both are monitored, all attacks crossing the shared edge would be observed by both of them and in this sense, they would overlap. Because of this overlap, it may not be wise to monitor both of them even if each of them individually has the greatest attack observe rate in the network.

4.3.3 NP-Completeness

To analyze the complexity of our problem, we convert the problem to its decision version: given the network topology and models as presented above, for k monitors, can we find a placement with at least p percent of the network wide attack observe rate? Here p is a constant.

First, we show this decision version problem is in NP. For a solution to this problem, the certifier adds up these k nodes' aor and then removes the overlapping portions to get the effective aor before verify whether the sum is at least p percent of the total. This can be done

www.manaraa.com

in polynomial time as showed in our Algorithm 3. So our problem belongs to the complexity class NP.

To show our problem is NP-complete, here we reduce the partial SET COVER problem [29] to our problem. The partial SET COVER problem can be described as: for a finite set U and a number of subsets of U, $S = \{S_1, S_2, ..., S_n\}$, find a partial cover $S^* \subseteq S$ that covers at least p percent of U with the minimum cardinality. Here S_d can be treated as the set of attack events that can be observed on relay node d and U is the set of all attack events in a unit of time in our problem.

Assume we have the polynomial-time algorithm that solves our placement decision problem. To solve the partial SET COVER problem, we generate a network such that the elements of U are represented as access nodes, i.e. $D = U$. For each $S_i \in S$, create a relay node with edges connecting it to the appropriate nodes in D. We then start from a random k and use the solution to find the k relay nodes that cover . If the union is larger than p percent of the total U, we decrease k; otherwise we increase k. Repeat the above procedure until the union is near equal to p percent of the total U. Because k is between 0 and the number of the relay nodes, the above procedure is in polynomial time. According to this reduction, if our problem can be solved, the partial SET COVER problem can also be solved. So our problem is NP-Complete.

4.4 Experiments with Heuristic and Greedy Solutions

4.4.1 Greedy Method in Polynomial Time

For the SET COVER problem, there is a well-known greedy algorithm [29] which has a $\ln N$ approximation where N is the cardinality of U. We modified that algorithm to fit our problem, Algorithm 4. In our case, S_r is the observed attack events set on relay node r and U is the set of all attack events in the network.

4.4.2 Heuristic Methods

As in previous chapter, we also provide heuristic solutions for this router level monitor placement problem. Here we use three heuristic methods: K-max, degree and betweenness.

www.manaraa.com

```
Input  : Attack set $S = \{S_r | r \in R\}$ and $U$
Output: Monitor placement $\hat{S} = \{S_{i1}, ..., S_{ik}\}$
begin
    $i \leftarrow 0$;
    while $U \neq \emptyset$ and $i < k$ do
        $max \leftarrow 0$;
        $i \leftarrow i + 1$;
        for $r \in R$ do
            if $S_r \neq \emptyset$ and $|S_r| > max$ then
                $max \leftarrow |S_r|$;
                $i^* \leftarrow r$;
            end
        end
        $U \leftarrow U \setminus S_{i^*}$;
        $\hat{S}_i \leftarrow$ the original set $S_{i^*}$;
        for $r \in R$ do
            $S_r \leftarrow S_r \setminus Si^*$;
        end
    end
    return $\hat{S}$;
end
```

Algorithm 4: Greedy Algorithm for Monitor Placement

K-max is to choose k nodes with the highest attack observe rates, due to the existing of overlapping, the sum of K-max may not be the maximized.

Degree and Betweenness are same as those in previous chapter except that here we only interested in the relay nodes.

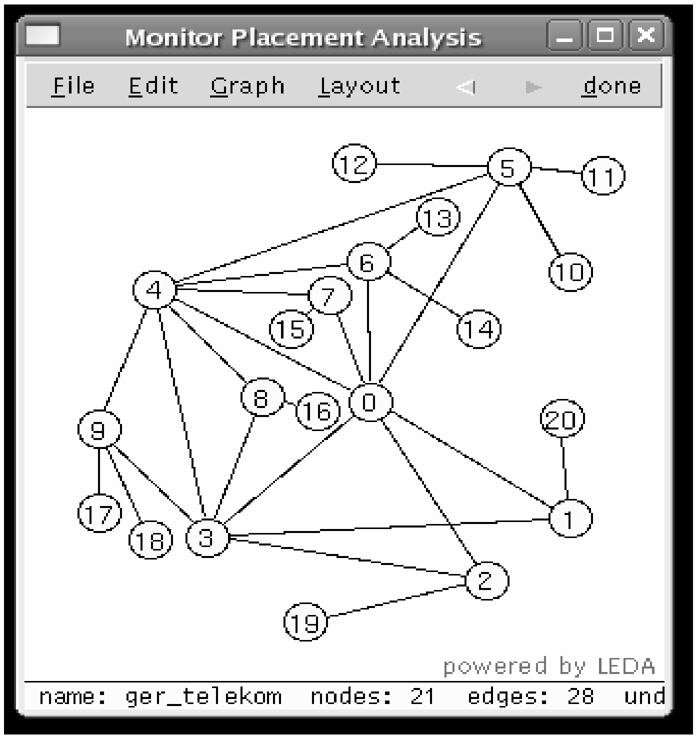
4.4.3 Experiment Results Comparison

Figure 4.3 is an example topology graph that has 11 access nodes and 10 relay nodes. The results based on the above four methods are showed in Figure 4.4.

According to these experiment results we found all methods can provide good optimization rates. And for this sample topology, greedy algorithm is a little better than heuristic methods K-max and Betweenness, but heuristic method Degree gives the best result.

If we consider algorithm complexity, we find that heuristic methods are better than the greedy algorithm. Here we assume the topology graph has n nodes and m edges. We know for method K-max, it computes attack observe rate for each relay node and needs traverse all

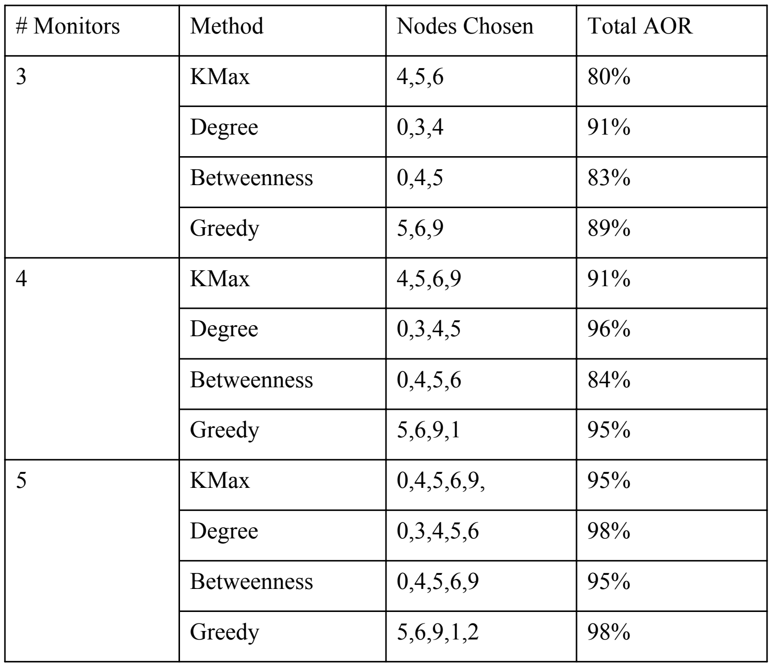
www.manaraa.com



name: ger_telekom nodes: 21 edges: 28 und

Figure 4.3 A Sample Network Topology

www.manaraa.com



# Monitors	Method	Nodes Chosen	Total AOR
3	KMax	4,5,6	80%
	Degree	0,3,4	91%
	Betweenness	0,4,5	83%
	Greedy	5,6,9	89%
4	KMax	4,5,6,9	91%
	Degree	0,3,4,5	96%
	Betweenness	0,4,5,6	84%
	Greedy	5,6,9,1	95%
5	KMax	0,4,5,6,9,	95%
	Degree	0,3,4,5,6	98%
	Betweenness	0,4,5,6,9	95%
	Greedy	5,6,9,1,2	98%

Figure 4.4 Results Compared

المنارة للاستشارات

www.manaraa.com

shortest path between all access nodes, and thus the complexity is O(n^3) if using the Floyd-Warshall algorithm; for method Degree, the complexity is O($n + 2m$) cause it only needs to visit all nodes and twice of all edges (the sum of degree is twice of the number of edges); for Betweenness, it runs BFS for all nodes and thus the complexity is O($n^2 + nm$). For the greedy algorithm, it first need to distribute attack events along all shortest paths and the complexity is O(n^3) and it also need do Set Minus operation which is O(n^4), so the total complexity is O(n^4). Furthermore, it also simulates attack events for each access node and thus needs O(n^2) space.

4.5 Node Correlation

Stepping stone correlation algorithms take the data recorded about two network sessions from different monitors and return a similarity score indicating their similarity. When there is an effective stepping stone correlation algorithm, monitors can utilize this algorithm to decide if two suspect links in fact belong to a same stepping stone chain. But one monitor alone can only correlate link data on its local storage, to find out the whole or as much as possible of the stepping stone chain, monitors need to work together. Hence, monitors will cooperate with each other to identify as many possible stepping stones for a given attack.

In POP or autonomous system (AS) level topology, each node can be an attack source, so the chain trace back procedure can be made node by node. This method is called the Direct Search Algorithm (DSA): one node receives a found attack pattern based on an incoming link and then correlates it with all of its outgoing links. If a match is found, continue to the next node. Figure 4.5 shows this algorithm.

For router level topology, DSA cannot work directly because monitors are located between two access nodes and one monitor may not have all data of an access node. In fact, one access node's incoming and outgoing link data are distributed in several monitors; and for one monitor, it may have several access nodes' data but not all because one monitor may be shared by several paths.

In our work, we assume monitors on the same routing path can talk to each other directly.

www.manaraa.com

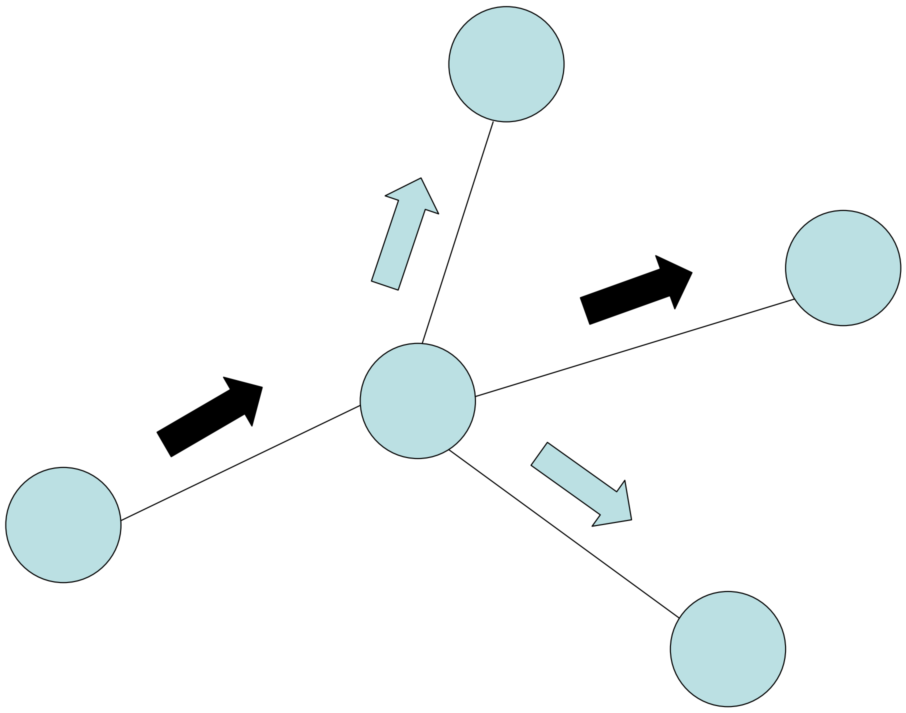


Figure 4.5 DSA for AS Level Topology

للاستشارات المنارة
www.manaraa.com

To quickly find out the chain as much as possible, we provide the following novel algorithm to do node correlation (node trace back):

- Start from a monitor M with an attack pattern Pst.
- M shares the Pst to all monitors that it can directly talk to.
- In parallel, the monitors get the pattern:
    - Correlate the pattern with their local data.
    - Report match.
    - Share the pattern to all other monitors that they can directly talk to except the monitor where the pattern came from.
    - If repeated pattern is received, just ignore.

4.6 Bi-directional Traffic Consideration

Asymmetric routing [43] is common phenomenon in Internet. Due to different routing policies in different parts of Internet, it is impossible to control end-to-end routing and paths traversed. So packets may not always traverse same downstream path as they do when forwarded upstream. This can be shown as Figure 4.6.

In some applications, it is important to acquire bi-directional traffic. But in general, monitors cannot be placed on data source or destination due to the security reasons, instead, they usually placed on mid-way routers as in our work. When this is done, asymmetric routing may cause monitors only capture one direction of data between two access nodes. In this section, we will discuss how to capture the bi-directional traffic via monitor placement. As mentioned before, we still use access node and relay node, but here they are no longer limited to routers but just refereing the traffic source, destination and transfer. And we also assume all routing information is given, so we know all the paths between any two access nodes.

المنارة للاستشارات
www.manaraa.com

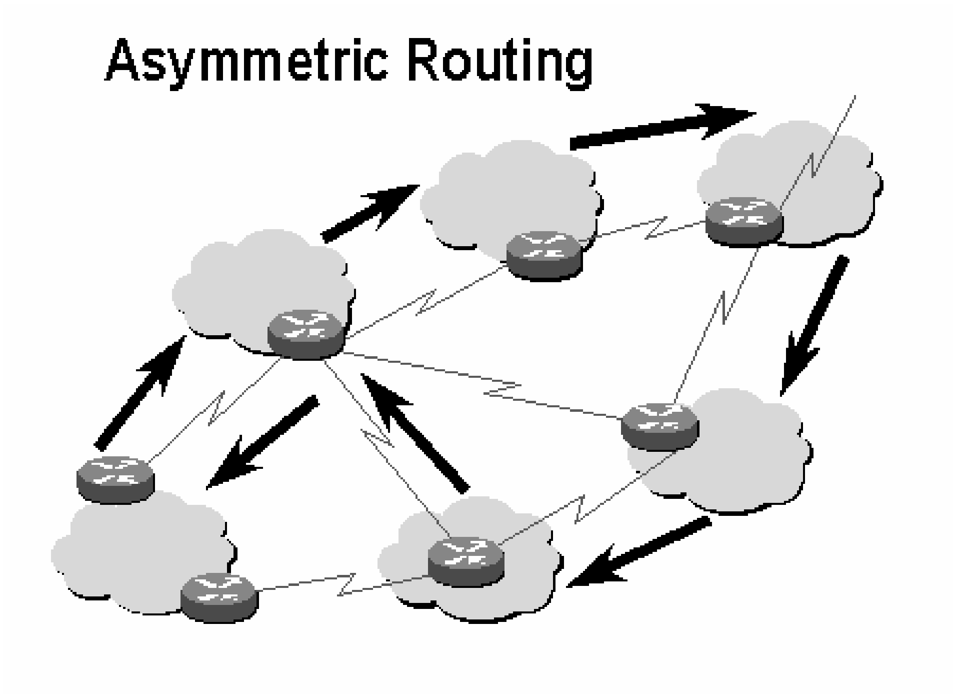


Figure 4.6 Asymmetric Routing

www.manaraa.com

4.6.1 Greedy Algorithms

It is clear that unless the forward path and the back path between an access node pair traverse a same monitor, one monitor can only get one direction of the data. We assume the monitors can correlate with each other, so we propose a straight forward idea that if there is a monitor on the forward path, we place another monitor on the back path. Based on this idea, we give greedy algorithms to place monitors to get bi-directional traffic data.

Before we describe our algorithms, we need define a metric to measure how much bi-directional traffic captured. In previous, we defined *aor* for a relay node to present how many attack events observed in a unit of time. In fact, this *aor* is the sum of attack traffic on the paths that pass through that relay node. And the sum of the *aor* is linear related to the number of paths covered. A path is covered in this context implies that the path passes through a node we will choose to place a monitor. So we call the sum of this *aor* as *total aor*, that means not consider the direction of the paths covered. We also define *bi-directional aor* as the following: if the paths of both directions are covered, *bi-directional aor* equals to the *total aor*, otherwise it is zero. For example, the network has path (a, b) with attack traffic x and path (b, a) with attack traffic y, if the monitoring system only covers the path (a, b), the *total aor* is x but the *bi-directional aor* is zeor; if both paths are covered, the *total aor* and the *bi-directional aor* are both x plus y. Generally, we are interested on the absolute value of *total aor* which stands for total coverage and how many percentage of *total aor* is *bi-directional aor*.

The first algorithm to cover bi-directional traffic we give is shown as the Algorithm 5, to simplify the problem, we assume attack traffic on each path is equal.

In this algorithm, function FindMAXNode is used to find the node that covers the most paths in a path set. We can found for covered paths, this algorithm will try it best to cover their corresponding reversed paths so as to ensure all captured traffic are bi-directional traffic. The problem of this algorithm is while to ensure a total percentage of *bi-directional aor*, the *total aor* may be low because of the low performance of the inner loop of the algorithm. To increase the *total aor*, we provide the second algorithm which is shown as Algorithm 6.

In this algorithm, function FindMAXNode is not only to find the node which covers the

www.manaraa.com

```
Input   : Network Topology with Routing Information
Output: k Nodes Chosen to Place Monitors
begin
    $A \leftarrow all\ \ paths\ \ between\ \ access\ \ nodes$;
    $B \leftarrow null$ and $C \leftarrow null$;
    $Monitor \leftarrow null$ and $numofm \leftarrow 0$;
    while $A \neq \emptyset$ and $numofm < k$ do
        $x \leftarrow FindMAXNode(A, Monitor)$;
        $numofm++$;
        $Monitor \leftarrow Monitor \cup \{x\}$;
        $pathAC \leftarrow \{paths \in A\ \ covered\ \ by\ \ x\}$;
        $A \leftarrow A \setminus pathAC$ and $C \leftarrow C \cup pathAC$;
        $pathBC \leftarrow \{paths \in B\ \ covered\ \ by\ \ x\}$;
        $B \leftarrow B \setminus pathBC$ and $C \leftarrow C \cup pathBC$;
        $pathAB \leftarrow \{paths \in A\ \ reversed\ \ with\ \ pathAC\}$;
        $A \leftarrow A \setminus pathAB$ and $B \leftarrow B \cup pathAB$;
        while $B \neq \emptyset$ and $numofm < k$ do
            $y \leftarrow FindMAXNode(B, Monitor)$;
            $numofm++$;
            $Monitor \leftarrow Monitor \cup \{y\}$;
            $pathBC \leftarrow \{paths \in B\ \ covered\ \ by\ \ y\}$;
            $B \leftarrow B \setminus pathBC$ and $C \leftarrow C \cup pathBC$;
            $pathAC \leftarrow \{paths \in A\ \ covered\ \ by\ \ y\}$;
            $A \leftarrow A \setminus pathAC$ and $C \leftarrow C \cup pathAC$;
            $pathAB \leftarrow \{paths \in A\ \ reversed\ \ with\ \ pathAC\}$;
            $A \leftarrow A \setminus pathAB$ and $B \leftarrow B \cup pathAB$;
        end
    end
    return $Monitor$;
end
```

Algorithm 5: Algorithm I for Bi-directional Traffic

www.manaraa.com

```
Input  : Network Topology with Routing Information
Output: k Nodes Chosen to Place Monitors
begin
    $A \leftarrow all\ \ paths\ \ between\ \ access\ \ nodes$;
    $B \leftarrow null$ and $C \leftarrow null$;
    $Monitor \leftarrow null$ and $numofm \leftarrow 0$;
    while $A \neq \emptyset$ and $numofm < k$ do
        $(x, p1) \leftarrow FindMAXNode(A, Monitor)$;
        $(y, p2) \leftarrow FindMAXNode(B, Monitor)$;
        if $p1 > rate * p2$ then
            $Monitor \leftarrow Monitor \cup \{x\}$;
            $pathAC \leftarrow \{paths \in A\ \ covered\ \ by\ \ x\}$;
            $A \leftarrow A \setminus pathAC$ and $C \leftarrow C \cup pathAC$;
            $pathBC \leftarrow \{paths \in B\ \ covered\ \ by\ \ x\}$;
            $B \leftarrow B \setminus pathBC$ and $C \leftarrow C \cup pathBC$;
            $pathAB \leftarrow \{paths \in A\ \ reversed\ \ with\ \ pathAC\}$;
            $A \leftarrow A \setminus pathAB$ and $B \leftarrow B \cup pathAB$;
        end
        else
            $Monitor \leftarrow Monitor \cup \{y\}$;
            $pathBC \leftarrow \{paths \in B\ \ covered\ \ by\ \ y\}$;
            $B \leftarrow B \setminus pathBC$ and $C \leftarrow C \cup pathBC$;
            $pathAC \leftarrow \{paths \in A\ \ covered\ \ by\ \ y\}$;
            $A \leftarrow A \setminus pathAC$ and $C \leftarrow C \cup pathAC$;
            $pathAB \leftarrow \{paths \in A\ \ reversed\ \ with\ \ pathAC\}$;
            $A \leftarrow A \setminus pathAB$ and $B \leftarrow B \cup pathAB$;
        end
        $numofm{+}{+}$;
    end
    return $Monitor$;
end
```

Algorithm 6: Algorithm II for Bi-directional Traffic

المنارة للاستشارات

www.manaraa.com

Algorithm-rate	One-directional	Percentage of Total	Percentage of Bi-directional
I	6499	64.40%	100%
II-1	9268	91.84%	98.34%
II-2	9238	91.54%	99.05%
II-3	9172	90.88%	99.30%

Table 4.1 Bi-directional Traffic

most paths in a path set, but also return the number of the paths that node covers. We provide a controllable parameter "rate" for this algorithm to adjust the tradeoff between *bi-directional aor* and *total aor*.

4.6.2 Experiment and Results

We use the data set collected by Active Measure Project (AMP) from NLANR [32] to evaluate our algorithm. This data set includes the traceroute data between 105 sites (after removing the sites unreachable in the data set). We use these traceroute paths as the routing paths to construct the network topology for our experiments. All the 105 sites will be treated as the access nodes and 1192 mid-way routers will be treated as the relay nodes. Table 4.1 is the results when 40 monitors placed.

According to the results, we can find algorithm I can ensure all traffic captured are bi-directional traffic, but the total coverage is low because the performance of the inner while loop is not good; algorithm II can gain a high total coverage but cannot ensure the 100% bi-directional coverage. With the increasing of the value of rate, total coverage decreases but percentage of bi-directional increases. The topology we used here is construct from traceroute data and it is not good to present a real network and cause the result is a little weird. In a real network, which should have more access nodes and the asymmetric routing is not so outstanding as this topology (100% asymmetric routing), algorithm I should have better results and algorithm II will not gain such a high bi-directional coverage.

المنارة للاستشارات
www.manaraa.com

4.7 Conclusion and Future Work

Using our approach, we can easily find reasonable locations to place monitors for POP level and router level topologies and also find a way to minimize the number of monitors deployed in a network while maintaining effective network attack attribution. Our greedy algorithms for bi-directional traffic is a new and effective approach to solve the problem of observing two-way traffic via monitor placement.

We address the remaining issues in our future work: First, because it is hard to find a network topology with routing information especially with asymmetric routing situation, we use traceroute data to simulate the topology in current experiment and found it was not very good to present a real network. So next we will do some research work for automatic topology generator which can create network topology with asymmetric routing information. Second, our threat model for router level topology is still simplistic, we will study the most threats for network and provide a more proper model. Third, we are going to continue work on the algorithms for bi-directional traffic capturing such as dynamically decide the rate by algorithm itself to gain the most proper tradeoff.

المنارة للاستشارات

www.manaraa.com

CHAPTER 5. ON SIMULATION OF ROUTING ASYMMETRY

In network security research, it is often assumed that the network topology and the routing information are given. But in fact, such data are difficult to get and this makes the research work lack of experiment data to do evaluation, just as the situation in our bi-directional traffic capturing research. In this chapter, we will present an approach to provide network topologies and routing information based on asymmetric routing simulation.

First, we will discuss the two challenges of doing the asymmetric routing simulation: proper topology generators and routing models. Then we will present our work based on different routing models.

5.1 Introduction

Internet is a collection of huge numbers of small networks which interconnect with each other. Most of these small networks are private and belongs to different entities. We call them autonomous systems, one network or sets of networks under a single administrative control. Based on this infrastructure, Internet routing can be categorized into intra-domain routing and inter-domain routing. In today's Internet, intra-domain routing usually uses RIP and OSPF, while BGP is the choice for inter-domain routing. Since different autonomous systems have different owners, in most cases, the inter-domain routing is less a technique problem but more related to the economic interests of who owns the network for the connection. For private issues, many policies are applied to inter-domain routing at the owner's will. Therefore, inter-domain routing can be called as policy-based routing. One common policy we will address in this chapter is the "Early Exit" policy.

Due to the different routing policies applied in different parts of Internet, it is impossible to

www.manaraa.com

control the end-to-end routing and paths traversed. The result is that packets may not always traverse the same downstream path as they do when forwarded upstream. This phenomenon in Internet is called asymmetric routing and has important influence on network research, such as our previous work about bi-directional traffic capturing [31]. We know for intra-domain routing, routing asymmetry is usually caused by the existence of multiple shortest paths; while for inter-domain routing, the "Early Exit" policy is responsible for asymmetric routing and it is also the dominant reason at a large scale and long term view of network.

In network research, we often assume that the network topology and routing information are given. However, such data is difficult to get, and even harder to be tailored for individual problem, especially when we need asymmetric routing information which is caused by routing policies such as "Early Exit". As a result, the research work is difficult to evaluate for lack of experiment data.

There are two ways to get network topology and routing information: network measurement and network simulation. The network measurement methods usually to record routing path through "traceroute" such as the Active Measurement Project (AMP) [32] or to record BGP routing tables to compute the possible routing paths such as the Route Views Project [33]. For our asymmetric routing information purpose, the data from network measurement has the following shortcomings:

- First of all, network measurement methods still cannot provide enough data set. Although the AMP and Route Views offered huge amount of data, the network infrastructure they measured seldom changes. As a result, the researchers could only get limited number of data instances of topology and routing information. For some network security research, the more data instances the better.

- Data based on "traceroute" is hard to be translated into network topologies, because "traceroute" only record interfaces. Since routers all have multiple interfaces, sometimes it is difficult to distinguish whether the downstream path and the upstream path pass through different routers or just pass through different interfaces on a same router.

المنارة للاستشارات
www.manaraa.com

- Data based on "traceroute", like AMP, is hard to present the real network situation for some research work. For example, AMP uses around 150 AMP monitors to do site-to-site measurements and approximately 1200 different intermediate nodes on the traceroute paths. If we treat AMP monitors as edge routers and intermediate nodes as core routers to construct the topology, and use traceroute paths as the routing paths, we will notice that the edge routers are much less than core routers in the network. Topology data with such characteristics is not good for network traffic capture research.

- Data from BGP routing table is hard to provide appropriate asymmetric routing information. BGP routing is a policy-based routing. Without knowing the policy, it is hard to construct topology and routing information to meet the asymmetric needs.

- Data based on "traceroute" like AMP does not provide any autonomous system level routing information and data from BGP routing table does not provide router level routing information.

In order to solve all these problems and provide data set for network researches which need network topology and routing (especially asymmetric routing) information, we presented some techniques to generate topology and routing data based on network simulation. The goal of this research is to provide data of large scale network topology with reasonable routing, especially asymmetric routing information.

There are two challenges for our work: network topology generators which are the source of the our network topology; and the routing models which are how is the "Early Exit" policy interpreted. In this chapter, the first part is to use power-law based network topology generator to generate both autonomous system level and router level topologies. Then mechanism to combine these two types of topologies together to present large scale networks is provided. Also, algorithms to simulate an completed "Early Exit" policy is given to achieve asymmetric routing. The second part will discuss how to create network topologies which can run the traditional "Early Exit" policy, the simulation algorithms are also provided.

The rest of this chapter is organized as the following: in section two, we will discuss the

المنارة للاستشارات
www.manaraa.com

challenges based on some related work about topology generation and routing simulation: first we will discuss some existed tools to do routing simulation and their inadequacies to achieve the research goal of this chapter; then we will discuss the previous work about network topology generation; and we also will discuss the "Early Exit" policy. In section three, we describe our problem; in section four, we present the the approach based on our full "Early Exit" policy and the prototype implementation, the experiment results evaluation are also presented; in section five, we discuss how to create the topology to simulate traditional "Early Exit" policy. And section six is the conclusions and some future work.

5.2 Challenges and Related Work

5.2.1 Routing Simulation Tools

To simulate IP level routing, there are two open-source tools that support both intra-domain routing and inter-domain routing (BGP-4): ns2 [39] and SSFNet [40]. Ns2 is a powerful network simulation tool that can do all levels of simulation for wired and wireless network. Standard ns2 only support intra-domain routing simulation and there is a BGP extension of ns2 called ns-bgp [41] which is ported from SSFNet by Tony Dingliang Feng to support the BGP simulation using ns2. SSFNet is a framework of open-source Java models of protocols (IP, TCP, UDP, BGP4, OSPF, and others), network elements (hosts, routers, links, LANs), and assorted support classes for realistic multi-protocol, multi-domain Internet modeling and simulation.

Consider our research goal, there are two primary limitations using these tools to do routing simulation:

- Performance issue: It is difficult for them to do large scale network routing simulation. In order to do the simulation with these tools, users have to describe each node and link of the topology graph using the tools defined languages. For large graph, it is very hard to do this manually. During the simulation, each node need set up a routing table reference to all other node. And that will cause long computation time and large memory usage.

www.manaraa.com

According to the author who implemented the distance vector routing algorithm in ns2, it cannot guarantee that the simulation will work for large scale network.

- Asymmetric routing issue: The most important issue to us, neither tool can simulate the "Early Exit" policy routing which is the dominant reason of the routing asymmetry.

5.2.2 Network Topology Generation

In our research, we use network topology generators as our data source. This give the first challenge to our work: what model shall we choose to create the network topology?

Many researchers believed that the tools based on power-law can present more realistic network characteristics.

Hongsuda Tangmunarunkit, Ramesh Govindan and Sugih Jamin [34] compared two measured network topologies, an autonomous system level one from the Route Views archive and a router-level one from some traceroute work, with the topologies generated by three categories of topology generators: random graph generators which is represented by the Waxman generator, structural generators which contains the Transit-Stub and Tiers generators, and degree-based (power-law) generators in which the simplest one is called the power-law random graph (PLRG) . Network topologies were compared in three basic metrics: the Expansion which is the rate of spreading, the Resilience which stands for the existence of alternate paths and the Distortion which describes the tree-like behavior of the network. Their results suggested that for large scale network topologies, degree-based generators capture the large-scale structure of the measured networks surprisingly well and are significantly better than structural generators; the hierarchy presented in the measured networks is looser and less strict than in the structural generators, and this characteristic is well captured by the hierarchical structure in degree-based generators.

The network topology generators based on power-law are Inet [35] from University of Michigan and Brite [36] from Boston University. Inet is a pure power-law generator, the nodes connection relation model can be described as the following: Let d_i and d_j be degrees of nodes i and j respectively, and $f(d_i)$ and $f(d_j)$ are the frequency of those degrees. Then ω_i^j is the

المنارة للاستشارات
www.manaraa.com

weighted value of d_j with respect to d_i and is defined as

$$\omega_i^j = MAX(1, \sqrt{(log\frac{d_i}{d_j})^2 + (log\frac{f(d_i)}{f(d_j)})^2}) \cdot d_j$$

Then $P(i,j)$, the probability that a node i with degree d_i connects to a node j of degree d_j is

$$P(i,j) = \frac{\omega_i^j}{\sum_{k\in G}\omega_i^k}$$

.

Brite is a multi-model generator, the power-law model it used is Barabasi-Albert model [37] which can be described as: When a node i joins the network, the probability that it connects to a node j already belonging to the network is given by

$$P(i,j) = \frac{d_j}{\sum_{k\in G} d_k}$$

Brite also has some features that closely related to our work, we will discuss them in the later section.

But some recent researches indicate the power-law in fact is not a proper model for network topology.

In [50], the authors thought the previous studies of topology including power-law have focused on interpreting measurements or on phenomenological descriptions and evaluation of graph-theoretic properties of topology generators. They proposed a complementary approach of combining a more subtle use of statistics and graph theory with a first-principles theory of router-level topology that reflects practical constraints and tradeoffs. The first type of constraints discussed in this work is router technology constraints: any router has a maximum number of packets that can be processed in any unit of time, and this causes a tradeoff between node degree and bandwidth. This constraint tell us the power-law is not very practical. The second type of constraints discussed is economic consideration which is even more important than technology consideration affecting the network design and deployment. The authors then presented some topology metrics which they thought are important to evaluate the network topologies: the first type is Commonly-used Metrics which are used by previous works; the second is Performance-Related Metrics which stands for technology consideration; and the third

www.manaraa.com

type of metrics is Likelihood-Related Metric which is mainly used to illustrate the difference among the topologies have the same degree distribution. This work thought while there is an inevitable tradeoff between model complexity and fidelity, a challenge is to distill from the seemingly endless list of potentially relevant technological and economic issues the features that are most essential to a solid understanding of the intrinsic fundamentals of network topology. This work then claims that very simple models that incorporate hard technological constraints on router and link bandwidth and connectivity, together with abstract models of user demand and network performance, can successfully address this challenge and further resolve much of the confusion and controversy that has surrounded topology generation and evaluation.

Also in [51], the authors re-examined the the BGP measurements that form the basis for the results reported in [38]. They found that by their very nature (i.e., being strictly BGP-based), the data provides a very incomplete picture of Internet connectivity at the AS level. The AS connectivity maps constructed from this data (the original maps) typically miss 2050% or even more of the physical links in AS maps constructed using additional sources (the extended maps). Subsequently, they found that while the vertex degree distributions resulting from the extended maps are heavy-tailed, they deviate significantly from a strict power law. Finally, they showed that available historical data does not support the connectivity-based dynamics assumed in [37]. Together, their results suggest that the Internet topology at the AS level may well have developed over time following a very different set of growth processes than those proposed in [37].

In [52] and [53], new models for AS level topology are discussed. The authors said: Two ASs are connected in the Internet AS graph only if they have a business "peering relationship". They developed a new optimization-driven model for Internet growth at AS provider-customer relationship level. The model's defining feature is an explicit construction of a novel class of intuitive, multi-objective, local optimizations by which the different customer ASs determine in a fully distributed and decentralized fashion their "best" upstream provider AS. Key criteria that are explicitly accounted for in the formulation of these multi-objective optimization problems are (i) AS-geography, i.e., locality and number of PoPs within individual ASs; (ii)

www.manaraa.com

AS-specific business models, abstract toy models that describe how individual ASs choose their "best" provider; and (iii) AS evolution, a historic account of the "lives" of individual ASs in a dynamic ISP market. They showed that the resulting model is broadly robust, perforce yields graphs that match inferred AS connectivity with respect to a number of different metrics, and was ideal for exploring the impact of new peering incentives or policies on AS-level connectivity.

Although these work show power-law model may be not proper for network topology generator, in this chapter we still use BRITE as our data source because lack of the generators based on these new models.

5.2.3 Early-exit Policy

In this chapter, we have two types of early-exit policy: full "Early Exit" and tradition "Early Exit".

The full "Early Exit" means the autonomous system will make a packet out of itself as soon as possible even without considering AS-level routing path. As showed in Figure 5.1.

The tradition "Early Exit" is defined in [42], which means when two ASs have multiple interfaces, the traffic will go through the interface which makes the traffic leave the AS as early as possible, but the AS-level routing path is unchanged. As showed in Figure 5.2.

5.3 Problem Description

In this section, we will discuss the background of this research and give a definition of the problem to be solved.

5.3.1 Problem Motivation

The motivation of the research work in this chapter is the lack of experiment data in our previous work about bi-directional traffic capturing via monitor placement. In that work, we need experiment data of network topologies with asymmetric routing information to evaluate our placement algorithms. But we found it is hard to get such data and through research we

www.manaraa.com

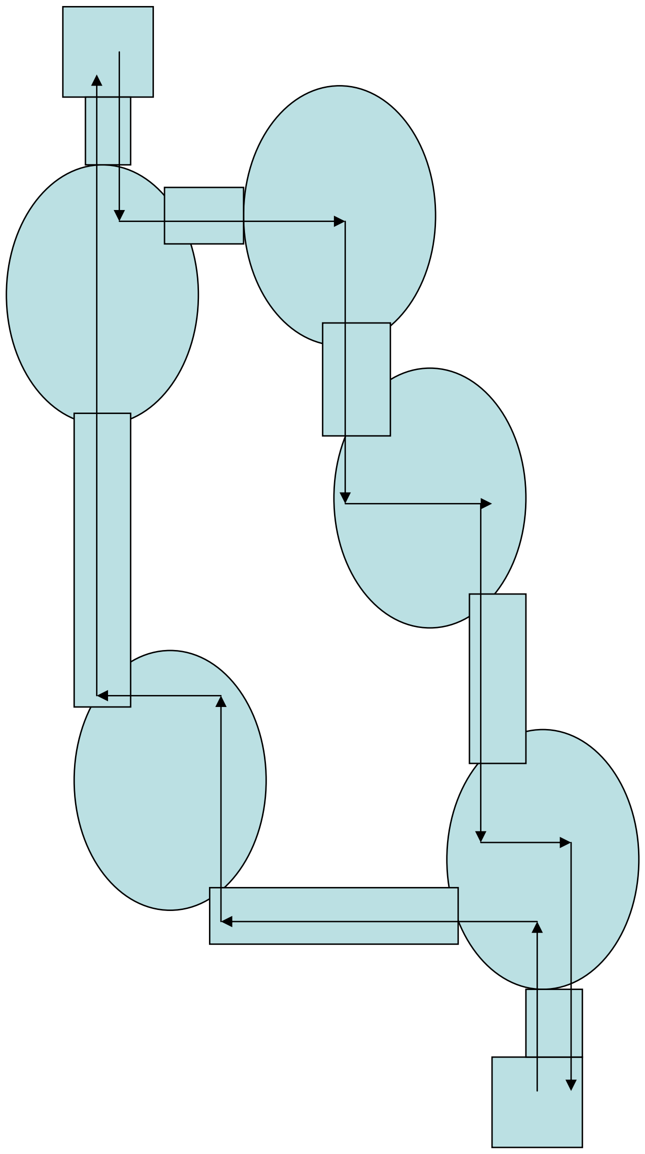


Figure 5.1 Full "Early Exit"

www.manaraa.com

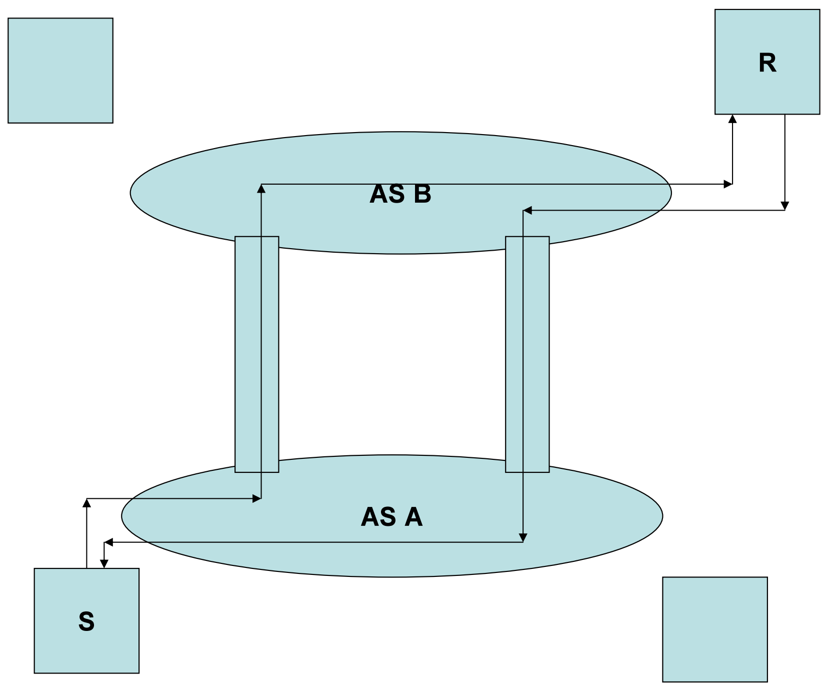


Figure 5.2 Tradition "Early Exit"

المنارة للاستشارات
www.manaraa.com

also found for many other research work related to network topology and routing, their source of experiment data are very limited.

5.3.2 Problem Definition

To solve these difficulty issues in obtaining the experiment data of large scale network topologies with asymmetric routing information, we define the following problem of asymmetric routing simulation: based on current research results about network topology generation, routing protocols and routing asymmetry, using the simulation method to simulate the policy routing to create data set of large scale network topologies with asymmetric routing information.

5.4 Approach Model

5.4.1 Model of the Approach

The model of our power-law based approach consists of two parts: two-level network topology generation, which includes border router selection, and the full "Early Exit" simulation.

5.4.1.1 Power-law Based Network Topology Generation

To create two-level (autonomous system level and router level) network topologies, we use a top-down method: First, we use Brite to create an autonomous system level topology; then, for each node in this topology, we use Inet to create a router level topology; at last, we use a border router selection algorithm to interconnect these router level topologies according to the autonomous system level topology. This top-down method is very straight-forward and also support in Brite.

The border router selection algorithms used in Brite is borrowed from GT-ITM [45]. We choose one called "Smallest degree non-leaf" to use in our implementation. The original "Smallest degree non-leaf" algorithm in Brite has a problem: it did not consider whether a node has already been chosen as a border router. The consequence of this problem is: in the final topology, an autonomous system will have very few number of border routers, in many cases,

www.manaraa.com

even only one. And thus cause our "Early Exit" simulation invalid because if there is only one "Exit", there is no meaning for "Early Exit". So we modified the original "Smallest degree non-leaf" algorithm in two ways: first, if a node has been chosen as a border router for n times, it will not be a candidate in the next n turns; second, after each selection, we add a new candidate node that its degree only larger than the degree of the existed candidate nodes. The detail is shown as Algorithm 7.

```
Input  : Source autonomous system $S$
Output: Border routers $Selected$
begin
    $S_{min}$ ←nodes of S with the smallest degrees but not leaf;
    $Selected \leftarrow null$;
    $Tmp \leftarrow null$;
    forall $v \in S$ do
        $Count[v] \leftarrow 0$;
    end
    while $not\ done$ do
        Move out any nodes in $Tmp$ which are timer off into $S_{min}$;
        $u \leftarrow$ random select from $S_{min}$;
        Move $u$ out of $S_{min}$;
        $Selected \leftarrow Selected \cup u$;
        $Count[u]++$;
        $timer\_start(u, Count[u])$;
        $Tmp \leftarrow Tmp \cup u$;
        $v \leftarrow$ A node not in $S_{min}$ and its degree only larger than nodes in $S_{min}$;
        $S_{min} \leftarrow S_{min} \cup v$;
    end
end
```

Algorithm 7: Border Router Selection

5.4.1.2 Full "Early Exit" Policy Routing Simulation

Now we have the final topology with both autonomous system level and router level information. Before we describe our "Early Exit" policy routing simulation algorithm, we first make some assumptions and give some definitions.

Assumptions:

- Instead of exchanging reachable information between border routers, we assume each border router knows all the paths from its autonomous system to the destination au-

المنارة للاستشارات
www.manaraa.com

tonomous system at the autonomous system level and all these paths are simple (no loop rule).

- For intra-domain routing, choose shortest path.
- All autonomous system including the sender apply "Early Exit" routing policy but may at different extent.

Definitions:

- **Full Selfishness**. The border router of an autonomous system applies "Early Exit" policy at the most extent, it does not care the whole length of a traffic packet will travel, it only ensure the packet travels the shortest length in its autonomous system but not violate the no loop rule.
- **Partial Selfishness**. The border router of an autonomous system will make a tradeoff between the whole length of a traffic packet will travel and the length the packet will travel in its autonomous system.
- **Selfishness Index**. A border router which is Partial Selfishness will select a number of autonomous system level paths that are as short as possible and then choose one of them to make the intra-domain routing length as short as possible. The number of the candidate paths is called the Selfishness Index. It is clear that the larger Selfishness Index, the more "selfish". In fact, the Full Selfishness is just the case while making all possible paths as candidate paths.

Based on the above assumptions and definitions, we describe our simulation algorithms as the following: For a Partial Selfishness incoming border router with Selfishness Index m, when it has traffic to forward, it first selects m autonomous system level paths as candidate paths based on a candidates selection algorithm which is shown as Algorithm 8, then based on these m candidate paths, it will decide the possible outgoing border routers. At last, the incoming border router will choose one of the possible outgoing border routers that is nearest to itself and send the traffic packet via intra-domain shortest path to that selected outgoing border router.

www.manaraa.com

If there are multiple nearest possible outgoing border routers, choose the one that is decided by the candidate path with the smallest length. For a Full Selfishness incoming border router, it choose the outgoing border router based on all possible autonomous system level paths to the destination to minimize the intra-domain routing length within its own autonomous system.

```
Input  : Selfishness index $m$, autonomous system level paths $G$, current autonomous
         system $s$, destination autonomous system $t$
Output: The set of candidate paths $P$ with size of $m$
begin
    $i \leftarrow 0$;
    $k \leftarrow 0$;
    $P \leftarrow null$;
    $SP \leftarrow null$;
    while $i < m$ do
        $k++$;
        $SP \leftarrow FindKthShortestPaths(G, s, t, k)$;
        $i+=|SP|$;
        $P \leftarrow P \cup SP$;
    end
    $P \leftarrow P[1..m]$;
    return $P$;
end
```

Algorithm 8: Candidate Paths Selection

The function $FindKthShortestPaths(G, s, t, k)$ in Algorithm 2 is the algorithm to find the k shortest paths from node s to node t in graph G. k is the rank of the algorithm, which means when $k = 1$, we look for the shortest paths; when $k = 2$, we look for the second shortest paths and so on. Because the K Shortest Path Problem [46] does not belong to the class P, to make problem simplify in our prototype implementation, we will provide an alternative algorithm called $FindKShortPaths$ which is modified from [47] instead of using the above candidate paths selection algorithm. Compare to that original algorithm in [47], our $FindKShortPaths$ will eliminate the repeated paths in the results and also have faster convergence.

5.4.2 Prototype Implementation and Experiments

Our prototype system has three parts: topology graph storage, routing simulation engine and routing asymmetry measurement.

المنارة للاستشارات

www.manaraa.com

5.4.2.1 Data Management

For each top-down topology instance, it has one autonomous system level graph file and several router level graph files. And the files generated by Brite and Inet will have different formats. To make data management clear, we use MySQL database to store the topologies. Table 5.1 to Table 5.4 are the database schemas we used in the prototype implementation. One thing need to mention here is in Table 5.2, the fields "srcrid" and "dstrid" are the two border routers that connect the two autonomous systems. These two fields and the the field "bgp" in Table 5.3 are filled by the border router selection algorithm mentioned above. The value of field "bgp" in Table 5.3 is the number of times of that router has been chosen as a border router.

Table 5.1 Node of Autonomous System

Field	Type	Description
tid	int	Id of an topology instance
asnum	int	Autonomous system number
linkid	int	Link Id
srcrid	int	Source router Id
dstrid	bool	Destination router Id

Table 5.2 Link of Autonomous System

Field	Type	Description
tid	int	Id of an topology instance
linkid	int	Link Id
srcas	int	Source autonomous system
dstas	int	Destination autonomous system
srcrid	int	Router Id in source
dstrid	int	Router Id in destination

5.4.2.2 Routing Simulation Algorithm

The implementation of routing simulation is shown as Algorithm 9 which assume the "Early Exit" policy is based on Partial Selfishness. In this algorithm, the functions *FindKShortPaths* and *FindShortestPath* are implemented using C++ and LEDA [48], the other part is imple-

www.manaraa.com

mented using Perl. For Full Selfishness scenario, the algorithm is similar except for using all possible paths as candidates instead of using the paths found by $FindKShortPaths$.

5.4.2.3 Asymmetry Metrics

To quantify the routing asymmetry, we use four metrics: two defined by ourselves and two borrowed from previous work [43] and [44].

The two metrics we defined are Absolute Match (AM) and Path Similarity (PS). Absolute Match is to measure whether the upstream path is the same as the conversed downstream path (match means they are same), it defines as: $AM = \frac{mp}{tp}$. mp means the number of matched paths pairs in the data set and tp means the total number of paths pairs in the data set. Path Similarity is the measurement of how likeness of the upstream path compared with the conversed downstream path, it defines as: $PS = \frac{2*ns}{np+nd}$. np means the number of nodes except source and destination in the upstream path and nd means the number of nodes except source and destination in the downstream path. ns is a little complex, it means the number of the same nodes in the same order appeared in the upstream path and in the the conversed downstream path. For example, if the upstream path is $\{s, a, b, k, c, y, d, m, t\}$ and the conversed downstream path is $\{s, a, b, x, l, d, z, c, p, t\}$, then $ns = 3$. A possible algorithm to compute ns is given in Algorithm 10.

The two borrowed metrics are Absolute Asymmetry (AA) which is the minimal composite dissimilarity and length-based Normalized Asymmetry (NA) defined as $NA = \frac{AA}{(LP+LD)}$. Here LP is the length of the upstream path and LD is the length of the downstream path.

Table 5.3 Node of Router

Field	Type	Description
tid	int	Id of an topology instance
asnum	int	Autonomous system number
rid	int	Router Id
rdegree	int	Degree of router
bgp	bool	If border router

www.manaraa.com

Table 5.4 Link of Router

Field	Type	Description
tid	int	Id of an topology instance
asnum	int	Autonomous system number
linkid	int	Link Id
srcrid	int	Source router Id
dstrid	bool	Destination router Id

```
Input  : Source AS $srcas$, source router id $srcrid$, destination AS $dstas$, destination
         router id $dstrid$
Output: Routing path from ($srcas$, $srcrid$) to ($dstas$, $dstrid$)
begin
    $interas \leftarrow srcas$;
    $inrid \leftarrow srcrid$;
    $RP \leftarrow null$;
    while $interas <> dstas$ do
        $ASPath \leftarrow ASPath \cup interas$;
        $nexthop \leftarrow FindKShortPaths(interas, dstas)$;
        $nexthop[1..k] \leftarrow LoopCheck(nexthop, ASPath)$; $candidaterid[1..k, 1..2] \leftarrow$
        $DatabaseQuery(nexthop[1..k])$;
        $distanceSP[1..k] \leftarrow FindShortestPath\ (interas, inrid, candidaterid[1..k, 1])$;
        Find $index$ that $distanceSP[index] = min(distanceSP[1..k])$;
        if $multiple\ index\ exist$ then
            Choose the index that minimize the AS level path;
        end
        $outrid \leftarrow candidaterid[index, 1]$;
        $RP \leftarrow RP\cup\ FindShortestPath(interas, inrid, outrid)$;
        $RP \leftarrow RP \cup ((interas, inrid), (nexthop[index], candidaterid[index, 2]))$;
        $interas \leftarrow nexthop[index]$;
        $inrid \leftarrow candidaterid[index, 2]$;
    end
    $RP \leftarrow RP\cup\ FindShortestPath(dstas, inrid, dstrid)$;
    return $RP$;
end
```

Algorithm 9: Partial Selfishness Routing Simulation

www.manaraa.com

```
Input  : Nodes list of the upstream path $NP$, nodes list of the conversed downstream
         path $ND$
Output: The number of the same nodes in the same order $ns$
begin
    $ns \leftarrow 0$;
    $TNP \leftarrow null$;
    $TND \leftarrow null$;
    for $i \leftarrow 0; i < |NP|; i++$ do
        if $NP[i] \in ND$ then
            $TNP \leftarrow TNP \cup \{NP[i]\}$;
        end
    end
    for $i \leftarrow 0; i < |ND|; i++$ do
        if $ND[i] \in NP$ then
            $TND \leftarrow TND \cup \{ND[i]\}$;
        end
    end
    $i \leftarrow 0$;
    $j \leftarrow 0$;
    while $i < |TNP|$ and $j < |TND|$ do
        if $TNP[i] == TND[j]|$ then
            $ns++; j++$;
        end
        $i++$;
    end
    return $ns$;
end
```

Algorithm 10: Same Nodes in the Same Order

المنارة للاستشارات

www.manaraa.com

5.4.2.4 Experiments Results

The experimental sample data produced by our prototype system that we will discuss here is a topology graph which has 250 autonomous systems and each autonomous system has a random number of routers between 125 to 375. According to the definition of Selfishness Index , we know the larger Selfishness Index, the "earlier exit" that an autonomous system will try to find. To analyze the effect of Selfishness Index on the routing asymmetry, we simulated 8000 pairs of routing paths on this graph for Selfishness Index 3 and 7.

The first result is the Absolute Match showed in Table 5.5. With Selfishness Index 3, we found more than seventy percent path pairs are symmetrical; but when Selfishness Index is 7, nearly half of the path pairs are asymmetrical.

Table 5.5 Compare Absolute Match

Selfishness Index	Match	Not Match	Asymmetry(%)
7	4344	4156	45.7
3	5768	2832	27.9

We then examine the distribution of asymmetry appeared in the data. Figure 5.3 to Figure 5.5 depict the distribution of Path Similarity, Absolute Asymmetry and length-based Normalized Asymmetry for Selfishness Index 3 and 7. From these results, we found more than half of the total 8000 paths pairs have large PS (between 0.9 and 1), small AA (0 and 1) and small NA (less than 0.05). We also found the trend in these three figures changed quickly which shows the larger asymmetry, the smaller chance to appear. These two characters mean for most of the routing paths pairs, the asymmetry is small. These characters are fit for the real Internet and have been verified by Yihua He in [43] and [44]. Our data have such characters provide the proof to show that our approach can produce reasonable asymmetric routing data.

By compare the results for Selfishness Index 3 and 7, we found larger Selfishness Index will cause larger routing asymmetry. In all three figures, the appearances of large asymmetry (small PS, large AA and large NA) for Selfishness Index 7 are much more than those for Selfishness Index 3. This character also has be verified in work [43] and [44]: the academic networks (presented by AMP dataset) have smaller routing asymmetry than commercial networks

المنارة للاستشارات
www.manaraa.com

(presented by RETRO dataset) because commercial networks are more "selfish".

5.5 Simulation Of Tradition "Early Exit"

Tradition "Early Exit" only happens when there are multiple interfaces between two autonomous systems. In this section, we will discuss how to create topology for it and then discuss the algorithm to simulate the Tradition "Early Exit".

5.5.1 Topology With Multiple Edges Between Two Nodes

To simulate tradition "Early Exit", we first need modify the topology generation to allow multiple edges between two nodes for AS-level topology graph. In our work, we use the following methods to achieve this:

```
Input  : AS-level topology graph $G(V, E)$
Output: AS-level topology graph $G'(V, E')$
begin
    $n \leftarrow |E|$;
    $ratio = 10$;
    $E' = null$;
    $L = null$;
    $i = 1$;
    while $int(n/ration) > 0$ do
        $i++$;
        $L = random(E - L, ration)$;
        $E' = E' + repeat(L, i)$;
        $ratio* = 10$;
    end
    return $E'$;
end
```

Algorithm 11: Multiple Edges

Function $random(E, k)$ means random select k edges from edge set E, function $repeat(L, i)$ means for each edge in set L, repeat i times and return the new edge set.

When we get the AS-level topology by Algorithm 11, we still use top-down method to create two-level topology graph and use Algorithm 7 to select the border routers.

المنارة للاستشارات
www.manaraa.com

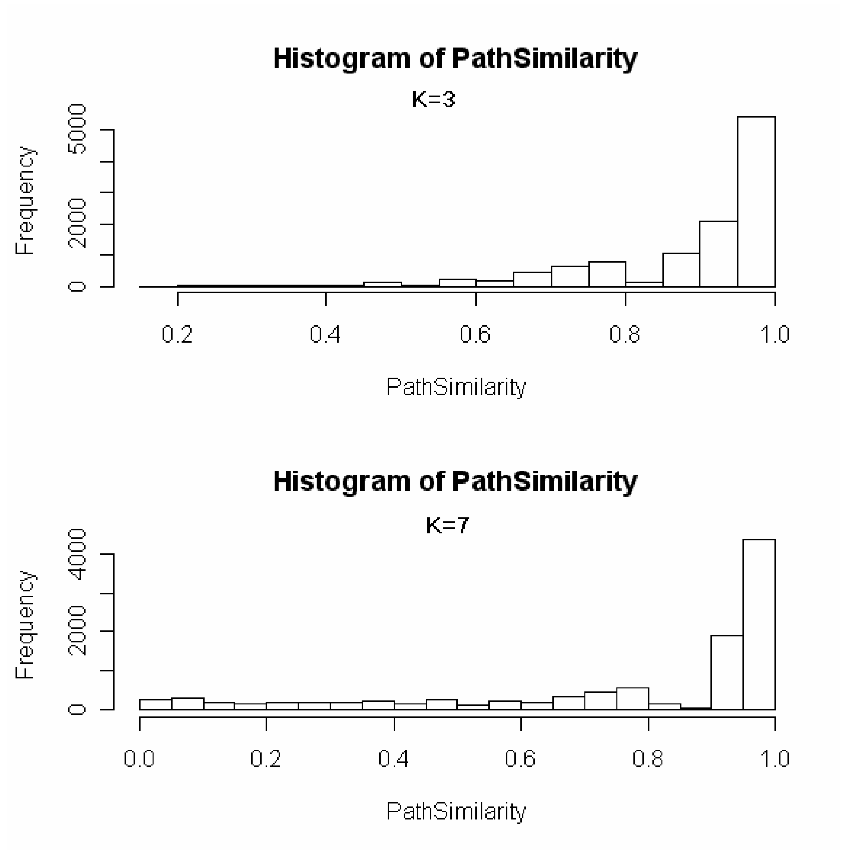


Figure 5.3 Compare Path Similarity Distribution

www.manaraa.com

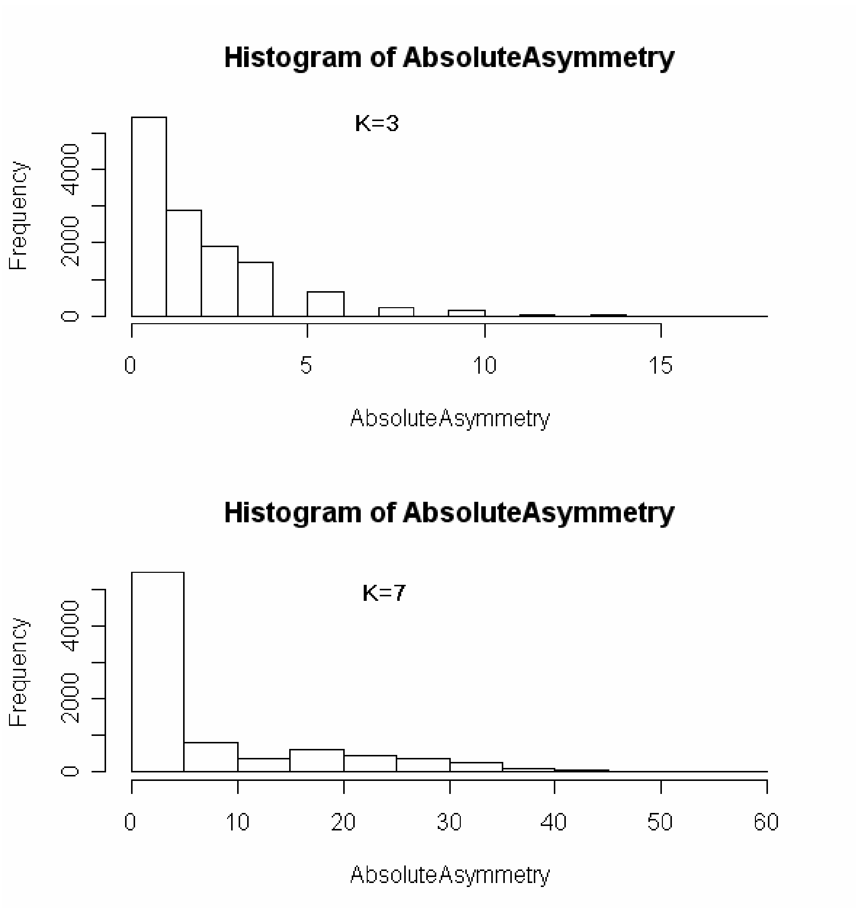


Figure 5.4 Compare Absolute Asymmetry Distribution

www.manaraa.com

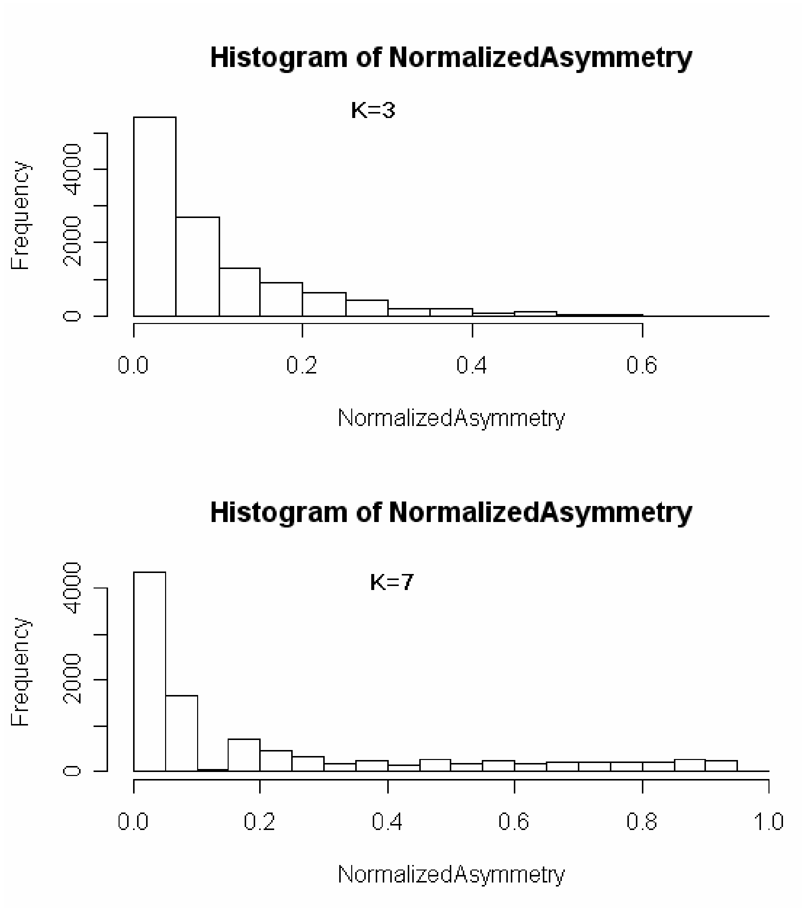


Figure 5.5 Compare Normalized Asymmetry Distribution

المنارة للاستشارات

www.manaraa.com

5.5.2 Tradition "Early Exit"

The algorithm to simulate tradition "Early Exit" is simple:

- First, use the shortest path as the routing path between source AS and destination AS. This determines the "path vector" in BGP, and the upstream "path vector" is same as the downstream "path vector" (but in reversed direction). So, at AS-level, there is no asymmetrical routing.

- Second, for AS X which is not the destination AS, based on AS-level routing path, we can find the next AS is Y. For the current node A in AS X, A is a border router if X is an intermediate AS or A is a internal router if X is the source AS, there may be multiple border router nodes connect to AS Y, choose the one nearest to A. This is the Candidate Path Selection for tradition "Early Exit". If AS X is the destination AS, then there is no choice, just select the shortest path from A to the destination node.

In Algorithm 12, function $Between(AS1, AS2)$ returns the border routers of $AS1$ that connect to $AS2$; function $ADJ(AS1, router1, AS2)$ returns the border router of $AS2$ connects $router1$ of $AS1$.

5.5.3 Experiments

We still use a topology graph which has 250 autonomous systems and each autonomous system has a random number of routers between 125 to 375. The AS-level topology has been modified by Algorithm 11 to generate multiple edges. And we still simulate 8000, 10000 and 12000 pairs of routing paths on this graph. We use the same metrics defined in the above section to evaluate the results.

The first result is the Absolute Match showed in Table 5.6. Compare to Full "Early Exit", the asymmetry rate is not very high for Tradition "Early Exit".

When check the distribution of asymmetry appeared in the data. Figure 5.6 to Figure 5.8 depict the similar trend of distribution of Path Similarity, Absolute Asymmetry and length-

www.manaraa.com

```
Input  : As-level topology G, Source AS $srcas$, source router id $srcrid$, destination AS
         $dstas$, destination router id $dstrid$
Output: Routing path from ($srcas$, $srcrid$) to ($dstas$, $dstrid$)
begin
    $interas \leftarrow srcas$;
    $inrid \leftarrow srcrid$;
    $RP \leftarrow null$;
    $ASPath \leftarrow FindShortestPath(G, srcas, dstas)$;
    $AsLen \leftarrow |ASPath|$;
    for $i \leftarrow 1; i < AsLen; i++$ do
        $interas \leftarrow ASPath[i-1]$;
        $nexthop \leftarrow ASPath[i]$;
        $candidates \leftarrow Between(interas, nexthop)$;
        $RLen \leftarrow MAX$;
        for $node \in candidates$ do
            if $|FindShortestPath(interas, inrid, node)| < RLen$ then
                $outrid \leftarrow node$;
            end
        end
        $RP \leftarrow RP \cup FindShortestPath(interas, inrid, outrid)$;
        $inrid \leftarrow ADJ(interas, inrid, nexthop)$;
    end
    $RP \leftarrow RP \cup FindShortestPath(dstas, inrid, dstrid)$;
    return $RP$;
end
```

Algorithm 12: Tradition "Early Exit" Simulation

www.manaraa.com

Table 5.6 Absolute Match

Total	Match	Not Match	Asymmetry(%)
8000	5697	2303	28.78
10000	6878	3122	31.22
12000	8343	3657	30.47

based Normalized Asymmetry comparing to the results when we simulate the Full "Early Exit". These characters of the results data are still fit Yihua He's results.

5.6 Conclusions and Future Work

In this work, we present an approach to provide large scale network topology with asymmetrical routing information by simulating "Early Exit" routing policy. By comparing the experiment result data produced by our prototype system with the results from other work, we found our data have the three characters that fit for the real Internet and thus the data are reasonable. So our approach achieved our research goal: provide an easy way to get reasonable large scale network topology with asymmetrical routing information.

During the implementation of our prototype system and experiments evaluation for the full "Early Exit", we found there are still three issues in our approach need to be improved. The first is about the border router selection which will greatly affect the simulation results. In the future we plan to investigate real topology data to find some statistical characteristics of BGP routers which can be applied in the border router selection algorithm. The second issue is the candidate paths selection algorithm used in the prototype system. Currently we use a simple but not real solution to the *Kth* Shortest Path problem. This algorithm works fine when Selfishness Index is small but may not find enough candidate paths when Selfishness Index increase. We should use some other algorithm for *Kth* Shortest Path problem in the implementation. The last point needs improvement is about "no path" processing. In our simulation algorithm, due to loop avoidance and Selfishness Index, there may be no path between some nodes pair. The current implementation just abort the routing simulation of such nodes pair. In the future, we need add processing to eliminate this "no path" scenario.

المنارة للاستشارات
www.manaraa.com

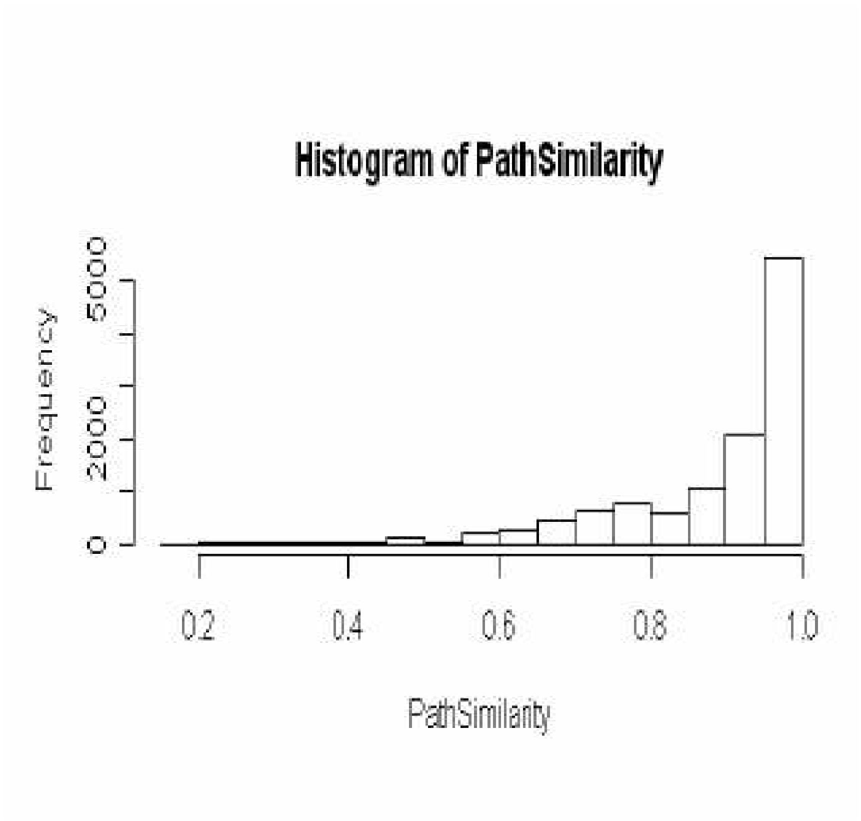


Figure 5.6 Compare Path Similarity Distribution

المنارة للاستشارات
www.manaraa.com

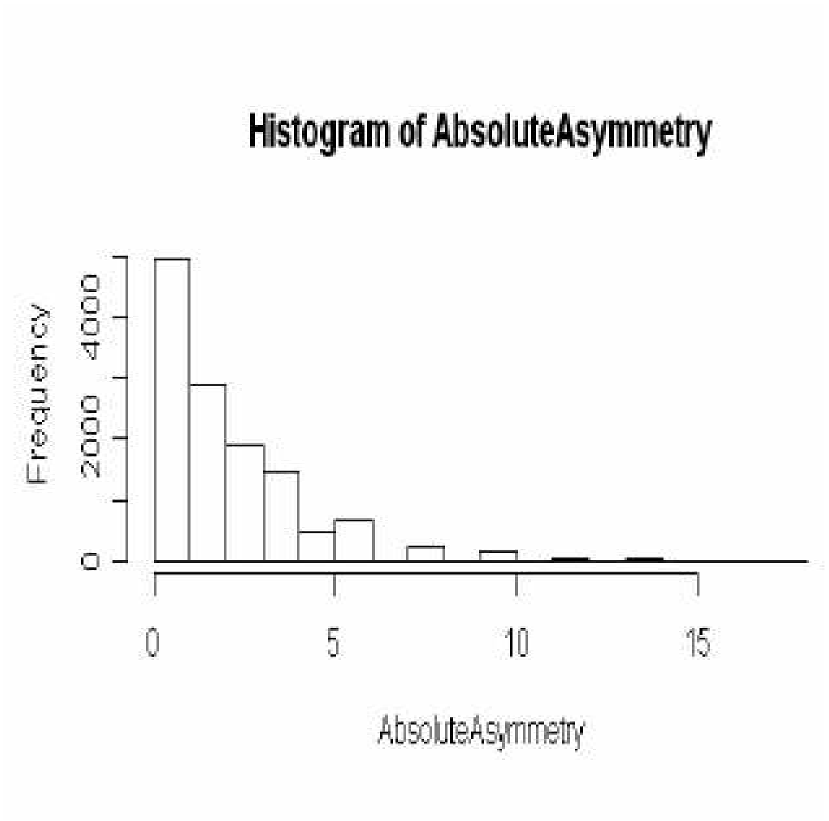


Figure 5.7 Compare Absolute Asymmetry Distribution

المنارة للاستشارات
www.manaraa.com

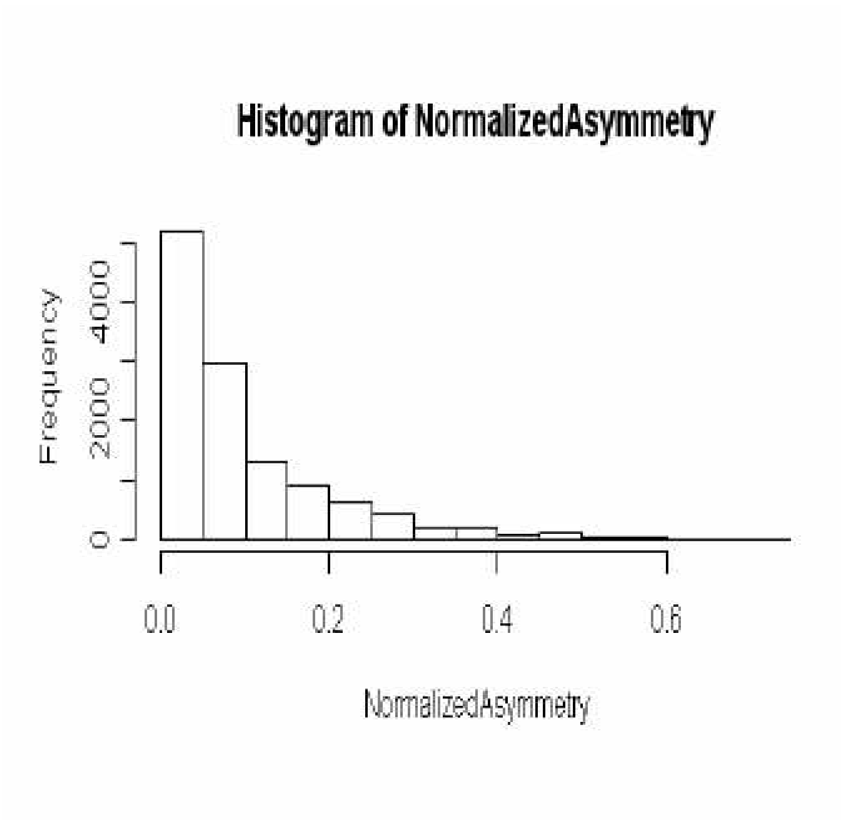


Figure 5.8 Compare Normalized Asymmetry Distribution

www.manaraa.com

Our asymmetrical routing simulation work is a whole new work and we still in the start phase. The techniques we used still need many improvements:

First of all, currently we use power-law based tools to create the topology, but in fact, the power-law seems be out-of-date. We need more appropriate model and tools to create more reasonable topology. And we may need use real topologies.

Second, the full "Early Exit" may not be existed in Internet. We need concentrate on tradition "Early Exit".

Third, there is no deterministic way to verify our result. Although we found our data fit the characters of asymmetrical routing tested in some previous work, but the fit is trend fit. It is hard to give an deterministic validation.

المنارة للاستشارات
www.manaraa.com

BIBLIOGRAPHY

[1] PCAP. [Online]. Available: http://www.tcpdump.org (Last accessed date: Nov. 2007)

[2] Kimmo Hatonen, and Jean Francois Boulicaut, and Mika Klemettinen, and Markus Miettinen, and Cyrille Masson. “Comprehensive Log Compression with Frequent Patterns.” [Online]. Available: http://liris.cnrs.fr/ jboulica/dawak03.pdf (Last accessed date: Nov. 2007)

[3] Arturo San and Emeterio Campos, “Arithmetic coding.” [Online]. Available: http://www.arturocampos.com/ac_arithmetic.html (Last accessed date: Nov. 2007)

[4] Kulesh Shanmugasundaram, and Nasir Memon, and Anubhav Savant, and Herve Bronnimann, “ForNet: A Distributed Forensics Network.” [Online]. Available: http://isis.poly.edu/projects/fornet/docs/talks/mmm-acns-2003-talk.pdf (Last accessed date: Nov. 2007)

[5] Alex C. Snoeren, “Single-Packet IP Traceback.” *IEEE/ACM Transactions on Networking (ToN), 2 Volume 10, Number 6, December 2002.* Pages 721-734.

[6] Yin Zhang, “Detecting Stepping Stones.” [Online]. Available: http://www.icir.org/vern/papers/stepping/ (Last accessed date: Nov. 2007)

[7] Thomas E. Daniels, “Reference Models for the Concealment and Observation of Origin Identity in Store-and-Forward Networks,” Ph.D. Dissertation. Purdue University, West Lafayette, Indiana. 2002. [Online]. http://clue.eng.iastate.edu/d̃aniels/papers/diss.pdf (Last accessed date: Nov. 2007)

المنارة للاستشارات
www.manaraa.com

[8] Yema Liverpool, "Monitor placement for network attack attribution using information theory metrics," masters thesis. [Online]. Available: http://archives.ece.iastate.edu/archive/00000114/01/Thesis_Final2.pdf (Last accessed date: Nov. 2007)

[9] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer, "Hash-Based IP Traceback," in *ACM SIGCOMM 2001*, San Diego, California, 2001.

[10] Georgios Rodolakis, Stavroula Siachalou and Leonidas Georgiadis, "Replicated Server Placement with QoS Constraints." [Online]. Available: http://genesis.ee.auth.gr/georgiadis/english/Publications/ConferencePapers /ReplicatedServerQoS.pdf (Last accessed date: Nov. 2007)

[11] Craig W. Cameron, Steven H. Low and David X. Wei, "High-Density Model for Server Allocation and Placement," in *SIGMETRICS 2002*. [Online]. Available: http://parapet.ee.princeton.edu/sigm2002/papers/p152-cameron.pdf (Last accessed date: Nov. 2007)

[12] Fred Buckley, Mary Lewinter. *A Friendly introduction to Graph Theory.* Prentice Hall. 2003.

[13] Marco Gaertler and Dorothea Wagner, "Algorithms for Representing Network Centrality, Groups and Density and Clustered Graph Representation," paper for *Coevolution and Self-organization in Dynamical Networks.* (COSIN) project. [Online]. Available: http://www.cosin.org/extra/deliverables/D6.pdf (Last accessed date: Nov. 2007)

[14] Ulrik Brandes, "A Faster Algorithm for Betweenness Centrality," in *Journal of Mathematical Sociology* 25(2):163-177. 2001.

[15] G. Sabidussi, "The Centrality Index of a Graph," in *Psychometrika*, 31, 58-603. 1996.

المنارة للاستشارات
www.manaraa.com

[16] J.M. Anthonisse, *The Rush in a Graph.* University of Amsterdam Mathematical Centre. Amsterdam. 1971.

[17] Oliver Heckman, “Topologies for ISP Level Network Simulation”. [Online]. Available: http://dmz02.kom.e-technik.tudarmstadt.de/heckmann/index.php3?content=topology (Last accessed date: Nov. 2007)

[18] C.E Shannon, “A Mathematical Theory of Communication,” in *The Bell System Technical Journey* Vol. 27, pp. 379-423, 623-656. 1948.

[19] Michael R. Garey, David S. Johnson, *Computers and Intractability: A Guide to the Theory of NPCompleteness*, 1979.

[20] C. Papadimitriou, “Worst-case and probabilistic analysis of a geometric location problem,” in *SIAM J. Comput.*, 10:542-557, 1981.

[21] O. Kariv and S. L. Hakimi, “An algorithmic approach to network location problems II: the p-medians,” in *SIAM J. Appl. Math.*, 37(3):539-560, December 1979.

[22] Linton C. Freeman, “A set of Measures of Centrality Based on Betweenness”, in *Sociometry* 40:35-41. 1977.

[23] Yin Zhang, “Detecting Stepping Stones,” [Online]. Available: http://www.icir.org/vern/papers/stepping/ (Last accessed date: Nov. 2007)

[24] Yu-Ju Kuo and Hans D. Mittelmann, “Facility location problem,” in *Interior Point Methods for Second-Order Cone Programming and OR Applications.* [Online]. Available: http://plato.asu.edu/papers/paper97/node5.html (Last accessed date: Nov. 2007)

[25] Rfc 2328. [Online]. Available: http://www.ietf.org/rfc/rfc2328.txt (Last accessed date: Nov. 2007)

المنارة للاستشارات
www.manaraa.com

[26] Benjamin Armbruster, J. Cole Smith, and Kihong Park, “A packet filter placement problem with application to defense against spoofed denial of service attacks.” [Online]. Available: http://www.cs.purdue.edu/nsl/sdarticle.pdf (Last accessed date: Nov. 2007)

[27] Jianqiang Xin et al, “An effective scheme for detecting stepping stone attacks in real-world networked systems.” Submitted to *InfoComm 2007.*

[28] L Qiu, V. Padmanabhan, and G. Voelke, “On the placement of web server replicas.” In *IEEE Infocom*, April 2001.

[29] Petr Slavik. “Approximation Algorithms for Set Cover and Related Problems,” PhD thesis, University of New York at Buffalo, Buffalo, NY, 1998.

[30] Yongping Tang, R. Yema. Liverpool, and Thomas E. Daniels, “Monitor placement for stepping stone analysis.” In *Proceedings of the 25th IEEE International Performance Computing and Communications Conference (Workshop on Information Assurance WIA-2006)*, Phoenix, AZ, April 2006.

[31] Yongping Tang and Thomas E. Daniels. “On the Economic Placement of Monitors in Router Level Network Topologies.” In *Proceedings of the 1st Workshop on the Economics of Securing the Information Infrastructure*, Washington DC, October, 2006.

[32] Active Measure Project. [Online]. Available: http://amp.nlanr.net/ (Last accessed date: Nov. 2007)

[33] University of Oregon Route Views Project. [Online]. Available: http://www.routeviews.org/ (Last accessed date: Nov. 2007)

[34] Hongsuda Tangmunarunkit, Ramesh Govindan and Sugih Jamin, “Network Topology Generators: DegreeBased vs. Structural,” in *Proceedings of the ACM SIGCOMM 2002*, Pittsburgh, PA, August, 2002.

[35] Inet Topology Generator. [Online]. Available: http://topology.eecs.umich.edu/inet/ (Last accessed date: Nov. 2007)

المنارة للاستشارات
www.manaraa.com

[36] BRITE: Boston university Representative Internet Topology gEnerator. [Online]. Available: http://www.cs.bu.edu/brite/ (Last accessed date: Nov. 2007)

[37] Albert-Lszl Barabsi, Rka Albert, "Emergence of scaling in random networks," in *Science*, pp. 509512, Oct. 1999.

[38] C. Faloutsos, P. Faloutsos, and M. Faloutsos, "On Power-Law Relationships of the Internet Topology," in *Proceedings of the ACM SIGCOMM*, Sept. 1999.

[39] The Network Simulator - ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/index.html (Last accessed date: Nov. 2007)

[40] Scalable Simulation Framework Network Models. [Online]. Available: http://www.ssfnet.org/homePage.html (Last accessed date: Nov. 2007)

[41] T. D. Feng, R. Ballantyne, and Lj. Trajkovic, "Implementation of BGP in a network simulator," in *Proceedings of the Applied Telecommunication Symposium ATS 04*, Arlington, VA, Apr, 2004. [Online]. Available: http://www.cs.sfu.ca/ ljilja/cnl/projects/BGP/ (Last accessed date: Nov. 2007)

[42] Iljitsch van Beijnum, *BGP Building Reliable Networks with the Border Gateway Protocol*, page 235, O'REILLY.

[43] Yihua He, Michalis Faloutsos, Srikanth Krishnamurthy and Bradley Huffaker, "On Routing Asymmetry in the Internet," in *Proceedings of the IEEE GLOBECOM 2005*, St. Louis, MO, Nov, 2005.

[44] Yihua He, Michalis Faloutsos and Srikanth Krishnamurthy, "Quantifying Routing Asymmetry in the Internet at the AS Level," in *Proceedings of the IEEE GLOBECOM 2004*, Dallas, TX, Nov, 2004.

[45] K. Calvert, M. Doar, and E. Zegura, "Modeling Internet Topology," in *IEEE Transactions on Communications, pages 160-163*, Dec, 1997.

المنارة للاستشارات
www.manaraa.com

[46] Myrna Palmgren and Di Yuan, "A Short Summary on K Shortest Path: Algorithms and Applications." [Online]. Available: http://www.esc.auckland.ac.nz/Mason/Courses/LinkopingColGen99/kth.pdf (Last accessed date: Nov. 2007)

[47] Implementation of K-shortest Path Dijkstra Algorithm used in All-optical Data Communication Networks. [Online]. Available: http://www.u.arizona.edu/ saurabh/SIE/kDijkstra546.pdf (Last accessed date: Nov. 2007)

[48] Algorithmic Solutions Software GMBH: LEDA. [Online]. Available: http://www.algorithmic-solutions.com/enleda.htm (Last accessed date: Nov. 2007)

[49] Kyoungwon Suh, Yang Guo, Jim Kurose, and Don Towsley, "Locating network monitors: complexity, heuristics, and coverage," in *Proceedings of the IEEE INFOCOM 2005.*

[50] Lun Li, David Alderson, Walter Willinger and AJohn Doyle, "First-Principles Approach to Understanding the Internets Router-level Topology," in *Proceedings of the ACM SIGCOMM 2004.*

[51] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, "The Origin of Power Laws in Internet Topologies Revisited," in *Proc. IEEE INFOCOM 2002.*

[52] H. Chang, S. Jamin, and W. Willinger, "Internet Connectivity at the ASlevel: An Optimization-Driven Modeling Approach," in *Proceedings of ACM SIGCOMM Workshop on MoMeTools*, August 2003.

[53] H. Chang, S. Jamin, and W. Willinger, "To peer or not to peer: Modeling the evolution of the Internet's AS-level topology," in *Proc. of IEEE INFOCOM 2006, Barcelona, Spain, 2006.*

المنارة للاستشارات

www.manaraa.com