



**INTERNET DENIAL BY HIGHER-TIER ISPS:  
A NAT-BASED SOLUTION**

BY

**ABDULAZIZ MUHAMMAD ALI AL-BAIZ**

A Thesis Presented to the  
DEANSHIP OF GRADUATE STUDIES

**KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**  
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the  
Requirements for the Degree of

**MASTER OF SCIENCE**

In

**COMPUTER ENGINEERING**

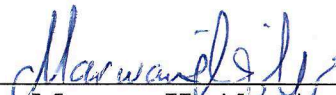
**JANUARY 2011**

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **ABDULAZIZ MUHAMMAD ALI AL-BAIZ** under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER ENGINEERING**.

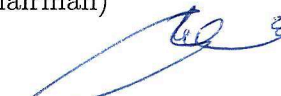
Thesis Committee



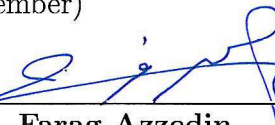
Dr. Marwan H. Abu-Amara  
(Chairman)



Dr. Ashraf S. Mahmoud  
(Co-chairman)



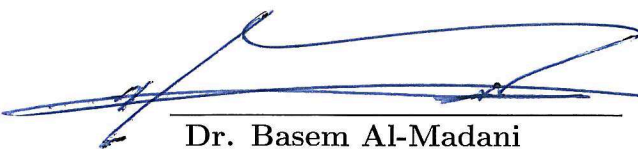
Dr. Mohammed H. Sqalli  
(Member)




Dr. Farag Azzedin  
(Member)



Dr. Alaaeldin A. M. Amin  
(Member)



Dr. Basem Al-Madani  
Department Chairman



Dr. Salam A. Zummo  
Dean of Graduate Studies

5/2/11  
Date

*Dedicated to  
my beloved parents and my dearest sisters*

# ACKNOWLEDGMENTS

*In the name of Allah, Most Gracious, Most Merciful*

All praise and gratitude be to Allah almighty, for giving me the courage and patience to accomplish this work.

My deepest appreciation goes to my thesis advisor, Dr. Marwan Abu-Amara, for his continuous help, invaluable guidance, and extreme patience throughout my thesis work. I owe special thanks to my thesis committee members, Dr. Ashraf Mahmoud, Dr. Mohammed Sqalli, Dr. Farag Azzedin, and Dr. Alaaeldin Amin, for all their help, support, and contributions throughout my study at KFUPM.

Acknowledge is also due to the King Fahd University of Petroleum and Minerals for providing me with the full support throughout the eight years I spent at it. I also would like to acknowledge King Abdulaziz City for Science and Technology (KACST) for funding this research.

# TABLE OF CONTENTS

<b>LIST OF TABLES</b>	<b>ix</b>
<b>LIST OF FIGURES</b>	<b>x</b>
<b>ABSTRACT (ENGLISH)</b>	<b>xiii</b>
<b>ABSTRACT (ARABIC)</b>	<b>xv</b>
<b>Chapter 1. INTRODUCTION</b>	<b>1</b>
1.1 Internet Resilience . . . . .	2
1.2 Internet Denial by Service Providers . . . . .	3
1.3 Research Motivation . . . . .	4
1.4 Thesis Objectives . . . . .	5
1.5 Thesis Contributions . . . . .	5
1.6 Thesis Organization . . . . .	6
<b>Chapter 2. INTENTIONAL INTERNET ACCESS DENIAL</b>	<b>8</b>
2.1 Internet Structure . . . . .	8
2.1.1 Internet: A network of networks . . . . .	8
2.1.2 Autonomous Systems . . . . .	9
2.1.3 Internet Service Providers . . . . .	9
2.1.4 ISPs Topology . . . . .	11

2.1.5	Border Gateway Protocol: The Glue of the Internet . . . . .	12
2.1.6	BGP Security Issues . . . . .	16
2.2	Related Work . . . . .	19
2.3	Internet Denial by Malicious ISPs . . . . .	20
2.3.1	What is Internet Denial . . . . .	20
2.3.2	Identifying Packet Source and Destination . . . . .	21
2.3.3	Motivations for Internet Denial . . . . .	22
2.3.4	Impact of Internet denial . . . . .	23
<b>Chapter 3. NAT-BASED SOLUTION TO INTERNET DENIAL</b>		<b>26</b>
3.1	Solutions to Internet Denial . . . . .	26
3.1.1	Control the Traffic Path . . . . .	27
3.1.2	Hide Traffic Identity . . . . .	28
3.2	Network Address Translation . . . . .	32
3.2.1	What is NAT? . . . . .	32
3.2.2	Address Translation Process . . . . .	34
3.3	NAT as a Solution to Internet Denial . . . . .	37
3.3.1	Deploying Gateway-Level NAT . . . . .	38
3.3.2	Private Network Configuration . . . . .	38
3.3.3	Local and Public DNS . . . . .	39
3.3.4	Design Scalability . . . . .	41
3.4	Performance Evaluation . . . . .	44
3.4.1	NAT Processing Delay . . . . .	45
3.4.2	Simulation of the NAT Solution . . . . .	48
3.5	Conclusion . . . . .	65
<b>Chapter 4. SERVER REACHABILITY BEHIND NAT</b>		<b>66</b>
4.1	NAT Impact on Server Reachability . . . . .	66
4.2	Related Work . . . . .	68
4.2.1	Multiple Websites on a Single Web Server . . . . .	69
4.2.2	Multiple Web Servers With a Single IP Address . . . . .	71

4.3	Proposed Solution for HTTP and SMTP Servers . . . . .	74
4.3.1	HTTP Servers Behind NAT . . . . .	74
4.3.2	SMTP Servers Behind NAT . . . . .	82
4.4	Management of IP addresses . . . . .	86
4.5	Performance Evaluation . . . . .	88
4.5.1	Modeling of Web Switch Delay . . . . .	89
4.5.2	Simulated Scenario . . . . .	90
4.5.3	Simulation Setup and Parameters . . . . .	91
4.5.4	Simulation of End-to-end Delay . . . . .	91
4.5.5	Simulation of Traffic Throughput . . . . .	95
4.6	Conclusion . . . . .	98
<b>Chapter 5. PEER-TO-PEER CONNECTIVITY BEHIND NAT</b>		<b>99</b>
5.1	NAT Impact on Peer-to-Peer Applications . . . . .	100
5.1.1	P2P vs Client-Server Applications . . . . .	100
5.1.2	Lack of End-To-End Connectivity . . . . .	100
5.1.3	Internet Denial Solution and P2P Applications . . . . .	101
5.2	Related Work . . . . .	102
5.2.1	NAT Behavior . . . . .	102
5.2.2	NAT Traversal Techniques . . . . .	105
5.3	Qualitative Analysis of NAT Traversal Techniques . . . . .	116
5.3.1	Analysis of Control-Based Techniques . . . . .	116
5.3.2	Analysis of Behavior-Based Techniques . . . . .	119
5.4	Usage of NAT Traversal for the Internet Denial Solution . . . . .	122
5.4.1	Deployment of Control-based NAT Traversal . . . . .	123
5.4.2	Deployment of Behavior-based NAT Traversal . . . . .	124
5.5	Performance Evaluation of NAT Traversal Using Relaying . . . . .	124
5.5.1	Simulation Scenario and Setup . . . . .	125
5.5.2	Simulation of End-to-End Delay . . . . .	126
5.5.3	Simulation of Traffic Bandwidth . . . . .	128

5.5.4	Conclusion . . . . .	129
<b>Chapter 6. DISCUSSION AND RECOMMENDATIONS</b>		<b>132</b>
6.1	Overview of the NAT-Based Solution . . . . .	132
6.1.1	Comparison of Different Solutions for Internet Denial . . .	132
6.1.2	Design Recommendations . . . . .	133
6.1.3	Performance Considerations . . . . .	134
6.1.4	Solution Limitations . . . . .	135
6.2	Case Study: Internet Denial in Saudi Arabia . . . . .	136
6.2.1	Internet Structure in Saudi Arabia . . . . .	136
6.2.2	Internet Denial . . . . .	137
6.2.3	Deployment of NAT Solution in Saudi Arabia . . . . .	138
<b>Chapter 7. CONCLUSION</b>		<b>141</b>
7.1	Summary of the Contributions . . . . .	141
7.2	Future Work . . . . .	142
<b>REFERENCES</b>		<b>144</b>
<b>VITA</b>		<b>156</b>



# LIST OF TABLES

2.1	Example of how BGP propagates reachability information . . . .	17
3.1	Network delay components, showing the processing delays . . . .	47

# LIST OF FIGURES

2.1	Organization of the three ISP tiers . . . . .	11
2.2	Visualization of Autonomous Systems on the Internet . . . . .	13
2.3	BGP views the Internet as a network of ASes . . . . .	15
2.4	Impact of malicious tier-3 ISP . . . . .	24
2.5	Impact of malicious tier-2 ISP . . . . .	25
2.6	Impact of malicious tier-1 ISP . . . . .	25
3.1	Outgoing packets sent through NAT . . . . .	35
3.2	Incoming response packets received by NAT . . . . .	37
3.3	DNS lookup for external servers . . . . .	40
3.4	DNS lookup for internal servers . . . . .	40
3.5	Initial design of the NAT solution . . . . .	41
3.6	Extended NAT design using pool of IP addresses . . . . .	42
3.7	Extended NAT design using multiple NAT routers . . . . .	44
3.8	G/D/c and G/D/1 queuing models . . . . .	49
3.9	Simulated scenario to measure the effect of NAT delay . . . . .	51
3.10	End-to-end delay for low UDP traffic . . . . .	53
3.11	End-to-end delay for medium UDP traffic . . . . .	53
3.12	End-to-end delay for high UDP traffic . . . . .	54
3.13	Increase of end-to-end delay for UDP traffic . . . . .	55
3.14	Relative increase of end-to-end delay for UDP traffic . . . . .	56
3.15	End-to-end delay for low TCP traffic . . . . .	57
3.16	End-to-end delay for medium TCP traffic . . . . .	57

3.17	End-to-end delay for high TCP traffic . . . . .	58
3.18	Increase of end-to-end delay for TCP traffic . . . . .	59
3.19	Relative increase of end-to-end delay for TCP traffic . . . . .	59
3.20	Throughput of high UDP traffic . . . . .	61
3.21	Throughput of high TCP traffic . . . . .	61
3.22	Relative decrease of throughput for TCP and UDP traffic . . . . .	62
3.23	Packet drop rate for high UDP traffic . . . . .	63
3.24	Packet drop rate for high TCP traffic . . . . .	64
3.25	Relative increase of packet drops for TCP and UDP traffic . . . . .	64
4.1	HTTP Request sent to a server with a public IP address . . . . .	67
4.2	HTTP Request sent to a server behind NAT . . . . .	68
4.3	The three steps of HTTP communication . . . . .	69
4.4	Usage of HTTP Host header in virtual hosting . . . . .	71
4.5	Initial setup for the solution of HTTP servers behind NAT . . . . .	75
4.6	Example of accessing an HTTP server behind NAT . . . . .	78
4.7	Load-balancing of web switches . . . . .	80
4.8	DNS is used to distribute the load over web switches . . . . .	81
4.9	SMTP protocol is used to deliver messages to email server . . . . .	83
4.10	Initial setup for accessing SMTP servers behind NAT . . . . .	85
4.11	Using the same IP addresses clients and servers . . . . .	87
4.12	Using separate IP addresses for clients and servers . . . . .	88
4.13	Simulated scenario for the HTTP server reachability solution . . . . .	90
4.14	End-to-end delay for low web traffic . . . . .	92
4.15	End-to-end delay for low web traffic . . . . .	93
4.16	End-to-end delay for high web traffic . . . . .	93
4.17	Increase in end-to-end delay for web traffic . . . . .	94
4.18	Relative Increase in end-to-end delay for web traffic . . . . .	95
4.19	Throughput for high web traffic . . . . .	96
4.20	Decrease of throughput for web traffic . . . . .	97

4.21	Relative decrease of throughput for web traffic . . . . .	97
5.1	Peer-to-peer and client-server application models . . . . .	101
5.2	Multi-level NAT affects connectivity in p2p applications . . . . .	102
5.3	Full Cone NAT . . . . .	103
5.4	Restricted Cone NAT . . . . .	104
5.5	Port-Restricted Cone NAT . . . . .	105
5.6	Cone and Symmetric NAT . . . . .	106
5.7	Control-based NAT traversal techniques . . . . .	107
5.8	Connection reversal technique . . . . .	110
5.9	Hole-punching technique . . . . .	112
5.10	Traffic relaying technique . . . . .	113
5.11	Multi-level NAT example . . . . .	117
5.12	Peers in multi-level NAT example . . . . .	119
5.13	Simulated scenario to measure the effect of relaying . . . . .	125
5.14	End-to-end delay for P2P network (logarithmic scale) . . . . .	127
5.15	Increase in End-to-end delay for P2P network (logarithmic scale) .	128
5.16	Relative increase in End-to-end delay for P2P network . . . . .	129
5.17	Throughput of P2P network with relaying . . . . .	130
5.18	Relative difference between relay's throughput and peer's throughput	130
6.1	Internet structure in Saudi Arabia . . . . .	137

# THESIS ABSTRACT

**NAME:** Abdulaziz Muhammad Ali Al-Baiz

**TITLE OF STUDY:** Internet Denial by Higher-tier ISPs: A NAT-Based Solution

**MAJOR FIELD:** Computer Engineering

**DATE OF DEGREE:** JANUARY 2011

Internet is an interconnection of independent Autonomous Systems (ASes). Most of the large ASes are operated by Internet Service Providers (ISPs), which are classified into 3 tiers based on their size and interconnections. Most of the Internet traffic is routed through the Internet core, represented by higher-tier ISPs. Because of the security flaws of Border Gateway Protocol (BGP), the presence of one or more malicious ISPs among the higher-tier ISPs can lead to many security concerns. Internet denial is when a malicious ISP blocks some or all the traffic that belongs to a specific network. The impact of Internet denial can be very critical. Network Address Translation (NAT) is used to design a solution that is scalable. In the NAT-based solution, outgoing traffic is address-translated into a non-blocked IP address in order to hide its identity. However, NAT limits end-to-end connectivity, causing servers within the victim network to become unreachable by external users. Application-layer information is used to design solutions for web and email server reachability behind NAT. NAT also limits peer-to-peer

(p2p) connectivity, preventing p2p applications from working properly. Existing solutions for NAT traversal are used to bypass this limitation. The impact of the proposed NAT-based solution on performance is negligibly small, and only a single NAT traversal technique, namely relaying, causes significant impact on the network performance.

# ملخص الرسالة

الاسم:

عبد العزيز بن محمد بن علي البيز

عنوان الرسالة: الحرمان المتعمد للوصول لشبكة الانترنت من قبل الفئة العليا بين

مزودي الخدمة: حل مبني على ترجمة عنوان الشبكة

التخصص:

هندسة الحاسب الآلي

تاريخ التخرج: يناير ٢٠١٠

تتكون شبكة الانترنت من العديد من الأنظمة المستقلة المتصلة ببعضها البعض. معظم هذه الأنظمة المستقلة، وخاصة الكبيرة منها، مملوكة لمقدمي خدمة الانترنت، والذين بدورهم ينقسمون إلى ثلاث فئات بناءً على حجمهم وطرق اتصالهم ببعض. تمر معظم حزم بيانات الانترنت من خلال نواة الانترنت، والتي يشكلها مقدمو الخدمة ذوي الفئات العليا. ونظراً لوجود العديد من الثغرات في بروتوكول بوابة الحدود (PGB) فإن وجود مقدم خدمة خبيث أو أكثر من ذوي الفئات العليا يشكل تهديدات أمنياً للبيانات. إحدى هذه التهديدات هو الحرمان المتعمد من الوصول لشبكة الانترنت، ويحدث ذلك عندما يمنع مقدم الخدمة مرور حزم البيانات المنتمية لشبكة ما من خلاله. تأثير من هذا التهديد قد يكون حاسماً جداً. الحل المقترح لهذه المشكلة يعتمد على استخدام أسلوب ترجمة عنوان الشبكة (NAT) لإخفاء هوية حزم البيانات المرسلة والمستقبلة عن مقدم الخدمة الخبيث، وبذلك يمكن إمرار هذه الحزم من خلاله بدون أن يستدل على مصدرها الحقيقي. إن استخدام هذا الأسلوب يؤدي لحل مشكلة الحرمان المتعمد لحزم البيانات الصادرة، ولكنه يؤدي إلى تقييد إمكانية استقبال الاتصالات الواردة، والذي يتسبب في مشاكل في الوصول إلى الخوادم الموجودة داخل الشبكة، ومحدودية الاتصال في البرنامج اللامركزية (الند للند) (P2P). يتم دراسة واقتراح حلول لهذه المشاكل وتقييمها من ناحية الأداء وقابلية التوسع، ويظهر من النتائج ان معظم الحلول المقترحة تسبب تأثيراً سلبياً ضئيلاً جداً.

## CHAPTER 1

# INTRODUCTION

The explosive growth of the Internet in the last decade has made it a very popular communication means in today's world. Many businesses have started to depend heavily on the Internet to conduct their daily tasks. An Internet disruption would not only be detrimental to businesses, public organizations and users, but it would also have catastrophic consequences on the economy. A report by Business Roundtable [1] estimates that the global economic cost of a major Internet disruption is approximately \$250 billion. Therefore, Internet resilience against such disruptions is very crucial. Design and implementation of solutions against potential Internet threats, whether they are accidental or intentional, is necessary to ensure the Internet robustness. This thesis tackles a very serious type of Internet disruption, namely the intentional blocking of Internet access by higher-tier service providers. This chapter introduces the idea of the targeted problem, together with the motivation and objectives of this thesis.



## 1.1 Internet Resilience

Many Internet major outages and disruptions took place during the last few years. In 2008, for example, three separate incidents of major damage to submarine optical cables caused Internet outage in Saudi Arabia and many countries in the Middle East and Asia [2–4]. Although the Internet disruption of these incidents seems regional, they have actually affected many businesses in other parts of the world [5].

Internet disruptions can happen at three different levels: the physical level, the routing level, and the application level. The damage of undersea cables falls under at the physical level. On the other hand, disruptions that may happen on the routing level include the accidental traffic blackholing by a small service provider in 1997 [6], which caused a significant portion of the Internet traffic to be dropped. Moreover, instances of some of the outages that can happen on the application level include the cases that affected some of the major major services, such as GMail [7] and Hotmail [8], as well as the Domain Name System (DNS) outages that has blocked access to many services [9, 10].

Such incidents raise questions about the Internet resilience against not only outages and misconfiguration, but also intentional attacks and denial of service. The impact of such Internet disruptions is not limited to the Internet-dependent businesses, but it also affects many other financial aspects. “An Internet meltdown would result in reduced productivity and profits, falling stock prices, erosion of consumer spending and potentially a liquidity crisis” [1].

## 1.2 Internet Denial by Service Providers

Internet Service Providers (ISPs) form the backbone of the Internet. They own large, worldwide networks to provide Internet connectivity to their customers. However, looking closely at how ISPs are structured to form the Internet, it can be seen that they have the most control over the Internet, at least in terms of connectivity and reachability. Large and medium-sized ISPs, often called tier-1 and tier-2 ISPs, respectively, are closer to the Internet core, and therefore, they carry most of the Internet traffic. The small and local ISPs, called tier-3 ISPs, are limited to carrying only traffic that belongs to their networks.

Because higher-tier ISPs control how the Internet traffic is routed, the presence of one or more malicious ISPs among them can lead to many security concerns. Traffic can be monitored, critical data can be exfiltrated, and packets can be modified. Even worse, traffic can be totally blocked from reaching the intended destinations.

The problem addressed by the thesis is related to the case when a malicious ISP, usually a tier-1 or tier-2 ISP, blocks some or all the traffic that belongs to a specific network. The victim network, which may range from a single user to an entire continent, will not be able to reach some portions of the Internet, specifically the networks that are accessible through the malicious ISP. We assume that the malicious ISP uses the Internet Protocol (IP) address to identify the source or destination of a packet, and drops that packet if it is either originated from or destined to the blocked victim network.

This thesis tackles the problem of blocking Internet access by ISPs by dropping traffic generated from or to be delivered to the victim network. The term *Internet Denial* is used throughout the thesis to address this problem.

### 1.3 Research Motivation

The impact of Internet denial depends on many factors, such as the locations and addresses of source and destination networks, the location and size of the malicious ISP, and the routing policies of the intermediate networks. In general, the malicious ISP has the capability to block access to networks it serves. In addition, other networks may be blocked because the malicious ISP is in their routing path. The worst case scenario could happen if one or more tier-1 networks conducted an Internet denial, as this will cause the victim network to be unable to reach most of the other networks on the Internet, causing a complete *Internet isolation*.

The victim network can be a single host, an enterprise network, or an entire country's network. The latter case is the most critical. As discussed earlier, an Internet outage could cause a stop to many sectors, especially with the increasing dependence on Internet availability.

The Internet in Saudi Arabia may become a victim of such type of problem, specially that none of the ISPs in Saudi Arabia is an international, higher-tier ISP. Therefore, a solution to increase Internet resilience against Internet denial at the routing level is very crucial to address. This thesis proposes such solution

that can circumvent the problem.

## 1.4 Thesis Objectives

The main objective of this thesis is to develop a solution for the problem of Internet denial at the routing level by higher-tier ISPs. The solution is meant to provide the maximum possible Internet availability with minimum changes in the network. The proposed solution is studied and investigated in terms of connectivity and performance, showing its advantages and disadvantages.

## 1.5 Thesis Contributions

To achieve the thesis objectives, the contributions of this thesis can be summarized as follows:

- Study the impact of Internet denial by higher-tier ISPs.
- Provide a solution that uses Network Address Translation (NAT), and analyze the related scalability issues.
- Evaluate the performance impact caused by the implementation of the proposed NAT solution using simulation.
- Study the limitations of the solution, such as the lack of end-to-end connectivity, and the unreachability of local services by external users; and provide

solutions for some of the services affected by these limitations, namely web servers, email servers, and peer-to-peer applications.

- Evaluate the proposed solutions for servers and p2p applications to measure their impact on the network performance.

## 1.6 Thesis Organization

The thesis is organized into the following chapters. Chapter 1.6 starts with a background about the Internet structure, the topology of ISPs, and the protocols that connect the Internet together. In addition, the problem of Internet denial is explored to see, based on the provided background, how serious and critical it is. The potential implications of the problem are also studied.

Chapter 2.3.4 starts with an exploration of different approaches to solve the Internet denial problem at the routing level. An approach that is based on Network Address Translation (NAT) is chosen to build the proposed solution on. The mechanisms of how the solution works, and how it is implemented, are described. Furthermore, we will look into the scalability issues of the solution, and discuss different possible design approaches. The performance of the solution is then evaluated using simulations to see its impact on the network performance.

One limitation of the proposed solution is the unreachability of local servers residing in the affected region, which is discussed in chapter 3.5. Solutions to the most common application servers, namely Web and Email servers, are proposed. Different scalability designs are also discussed to balance the load and improve

the performance. Simulations are performed in order to evaluate the performance impact on servers.

The following chapter, chapter 4.6, discusses another critical connectivity limitation of the proposed solution. The solution prevents many peer-to-peer applications from establishing connections correctly. Therefore, a survey of the existing NAT traversal techniques is presented, and the impact of the solution on peer-to-peer applications, both on performance and connectivity, is discussed.

Chapter 5.5.4 presents a discussion of the complete proposed solution in terms of connectivity, performance, and limitations. It also presents an example of implementing the solution on the ISPs in Saudi Arabia.

The thesis finally concludes in chapter 6.2.3, where the overall picture of the proposed solutions is summarized, with a list of potential improvements and unresolved issues as future work.

## CHAPTER 2

# INTENTIONAL INTERNET ACCESS DENIAL

In this chapter, the problem of *Internet Denial* is discussed in more details. Some background on the Internet structure and connectivity is presented in order to provide a better understanding of the likelihood and impact of this problem.

## 2.1 Internet Structure

### 2.1.1 Internet: A network of networks

The global Internet is a system of interconnections between a large number of separate, heterogeneous networks. These networks use a unified set of standard protocols to communicate between them. The Internet is a *network of networks* that consists of millions of private and public, business, academic, and government networks. Each of these networks is run and administrated independently, and therefore, these networks are usually called *Autonomous Systems*.

## 2.1.2 Autonomous Systems

An Autonomous System (AS) is a network of hosts, routers and IP prefixes that run under a single administration, and have a single and clearly defined external routing policy [11]. An AS may have different routing protocols and policies within it, but it must have only a single policy when it interacts with other ASes. ASes on the Internet are of different sizes; they range from a single router to a cross-continent network. Most of the large ASes are owned by Internet Service Providers (ISPs). An ISP's network is ideally one AS, since the network is under a single administrative entity. However, many ISPs have more than one AS in their networks.

## 2.1.3 Internet Service Providers

Internet Service Providers (ISPs) are companies that offer their customers access to the Internet. An ISP connects to its customers using some data communication technology, and provides them with the service of delivering their Internet datagrams. All ISPs on the Internet are interconnected, either directly, via physical links, or indirectly, via intermediate ISPs.

### Interconnection Between ISPs

Before discussing the 3 tiers of ISPs, let us first look into how ISPs are interconnected. The two types of relationships between ISPs are Transit and Peering.

*Transit* interconnection is a provider-customer relationship. It is simply when



an ISP sells dedicated access to its customer ISPs via private leased-line circuits. The customer ISPs pay for the Internet access in this type of interconnection.

*Peering*, on the other hand, refers to a interconnection between two ISPs to exchange traffic for the mutual benefit of both parties. Each ISP provides the other ISP with access to its networks and customers' networks. This interconnection does not involve payments for the access service, and hence, it is sometimes called "settlement-free peering" to reflect the fact of cost-free interconnection. There are two types of peering, depending on the physical connections that are used: *private peering*, where a point-to-point link is used to physically connect the two ISPs, and *public peering*, where multiple ISPs are interconnected at an *Internet Exchange Point* (IXP) using a shared switch fabric.

### **Classification of ISPs**

ISPs are usually classified into 3 *tiers*. The classification can be based on their size, their type of interconnections, or the type of services they provide and customers they serve. Although there is no clear definition of the boundaries of these classifications, the most common definition is based on how an ISP is interconnected to other ISPs.

Tier-1 ISPs are networks that can reach every other network on the Internet solely via peering, without purchasing transit or paying any settlements. Based on this definition, tier-1 ISPs are likely to form a full-mesh, i.e., each tier-1 ISP is interconnected to all other tier-1 ISPs. Tier-1 ISPs usually own large networks that cover one or more continent.

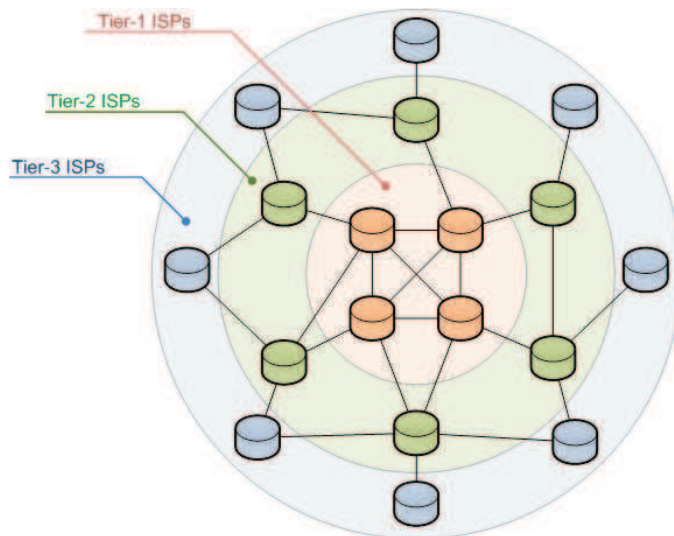


Figure 2.1: Organization of the three ISP tiers

Tier-2 ISPs are networks that peer with other ISPs, but still pay for transit interconnections to reach some portions of the Internet. They usually peer with other tier-2 ISPs, and purchase transit from one or more tier-1 ISP. Tier-2 ISPs are smaller than tier-1 ISPs, as they mostly cover one or few countries.

Tier-3 ISP is a network that “solely purchases transit from other networks to reach the Internet” [12]. Thus, tier-3 ISPs typically connect to one or more tier-2 and tier-1 ISP in order to reach the Internet. Tier-3 ISPs are relatively small, covering a country or a metropolitan area of a country. Therefore, tier-3 ISPs usually provide Internet service to end-users.

#### 2.1.4 ISPs Topology

The classification of ISPs based on their interconnections results in forming a hierarchy for the Internet. Figure 2.1 shows how ISPs are organized into three

tiers. Tier-1 ISPs form the Internet core, tier-3 ISPs form the Internet edge, and tier-2 ISPs are in between them. However, the number of ISPs world-wide has rapidly increased; more interconnections are formed between them, and the structure started to lose its hierarchical topology. Today's Internet consists of more than 30,000 interconnected AS [13]. Figure 2.2 shows a visualization of these ASes, organized such that the AS with more interconnections are closer to the center of the figure. It can be seen that a small portion of these networks belong to higher-tier ISPs, representing the Internet core. The remaining ASes, the ones closer to the Internet edge, belong to tier-3 ISPs and end-user networks.

### 2.1.5 Border Gateway Protocol: The Glue of the Internet

The Internet consists of extremely heterogeneous networks. These networks use different routing protocols, and different physical networks. In order to communicate with each other, all of them use the same set of standard protocols, the TCP/IP suite. These protocols include the network-layer *Internet Protocol* (IP), the transport-layer *Transmission Control Protocol* (TCP) and *User Datagram Protocol* (UDP), and a set of standard application-layer protocols, such as *Hypertext Transfer Protocol* (HTTP), *File Transfer Protocol* (FTP), and *Domain Name System* (DNS).

For two hosts to communicate with each other, they need to use the same application- and transport-layer protocols. The networks between them should use the IP as a network-layer protocol. Given this setup, the end systems can

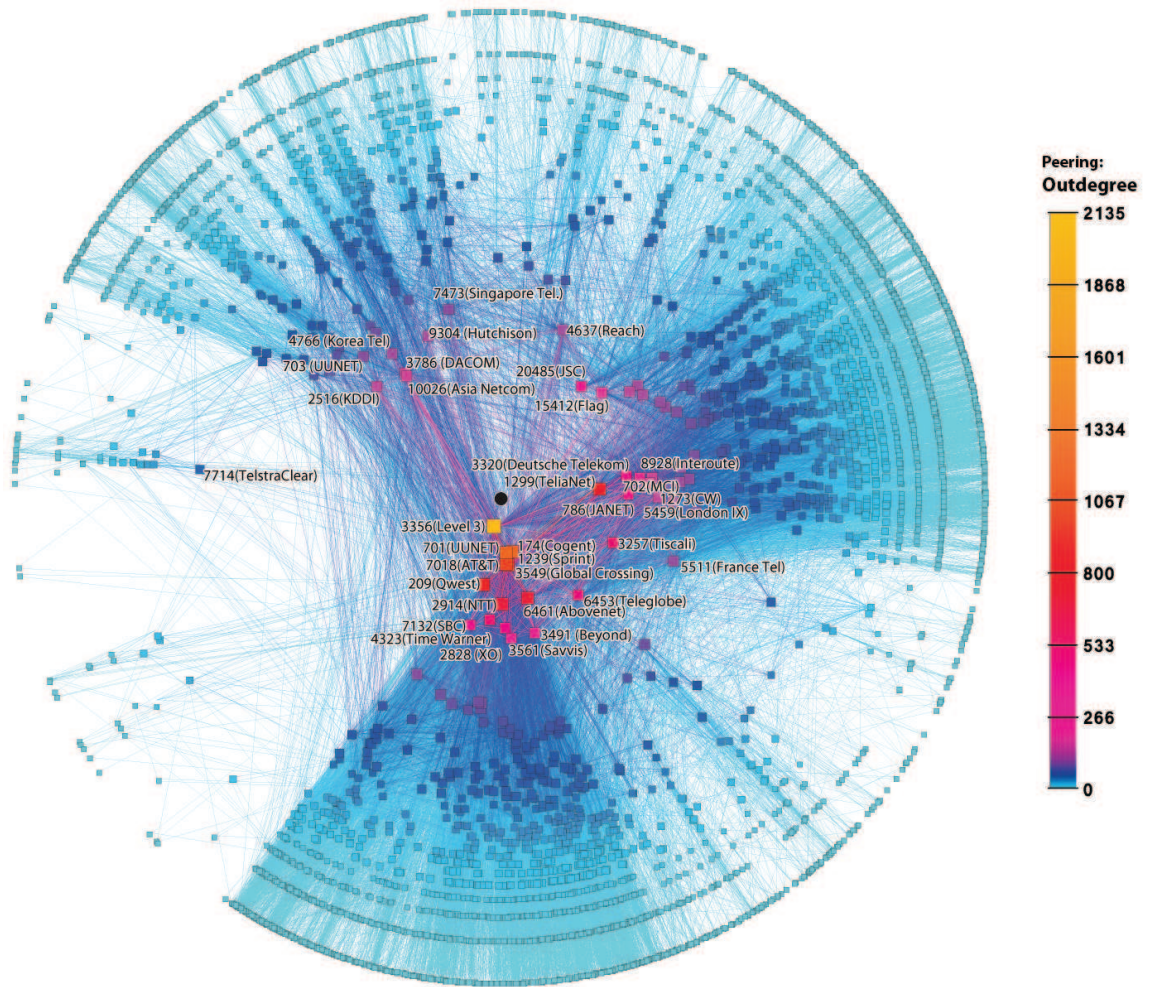


Figure 2.2: Visualization of Autonomous Systems on the Internet

communicate. However, when there are many other hosts on the Internet, and those hosts are on different networks, a routing protocol is needed to provide reachability information for a given destination. The routing protocols that are used within a network, such as *Routing Information Protocol* (RIP), and *Open Shortest Path First* (OSPF), are called *Interior Gateway Protocols*. They are only used to provide routing within an AS, and they normally try to optimize the network performance.

When two ASes are interconnected, they use an *Exterior Gateway Protocol*. The Internet uses *Border Gateway Protocol* (BGP) as the standard inter-AS routing protocol. All networks on the Internet use BGP to exchange routing information between each other.

BGP maintains a table of IP subnets, also referred to as *IP prefixes*, together with their reachability information over ASes. BGP is different than other routing protocols as it does not view the network as hosts and routers. Rather, it sees the Internet as one network of *atomic* ASes, as shown in figure 2.3. Because of the business nature of ISP interconnections, BGP is a policy-based routing protocol; it accepts, denies, and optimizes routing paths based on policy, not performance.

BGP provides each AS a means to collect reachability information from neighboring ASes, determine the best paths to IP prefixes based on reachability information and the predefined AS policy, and propagate reachability information to all routers within the AS.

Establishing a BGP session between two ASes requires a direct interconnection

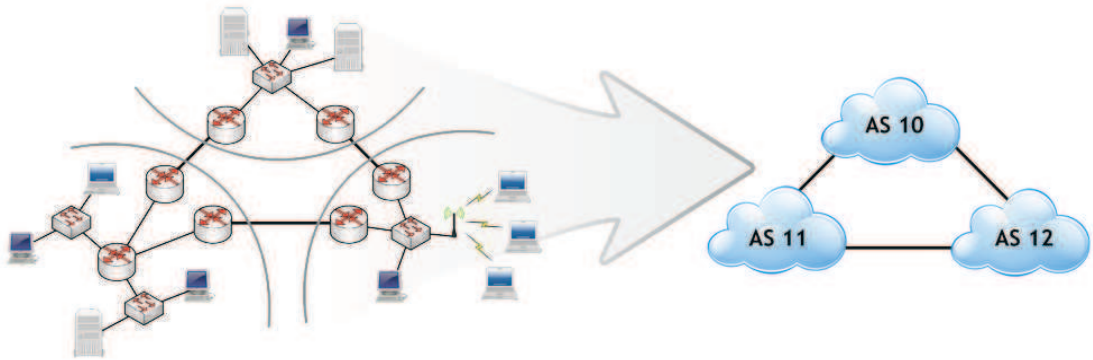


Figure 2.3: BGP view of the Internet as a set of interconnected Autonomous Systems

between them. BGP runs over TCP to achieve reliability. When two ASes are connected via BGP, the complete routing table is exchanged once, then only route changes are sent through update messages. Each AS performs the following steps:

1. Advertise the local IP prefixes to neighboring ASes
2. Obtain prefixes that are advertised by neighboring ASes
3. If there are more than one path to a given prefix, a selection process is initiated to choose the *best* path.
4. Advertise the selected paths to neighboring ASes, except to the AS that sent it.

Because BGP is policy-based, an AS can define policies to control which prefixes are advertised, which neighboring ASes receive those advertisements, and which incoming advertisements are accepted.

An AS may receive several advertisements for the same prefix. Hence, a path-selection process is performed to select the best path. The selection process de-

depends on the path length, policy, and other factors. Only the selected path is advertised to neighbor ASes.

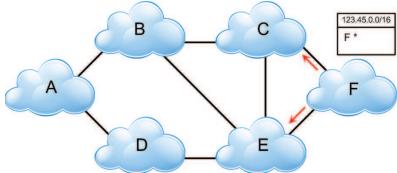
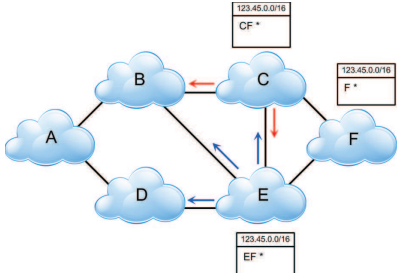
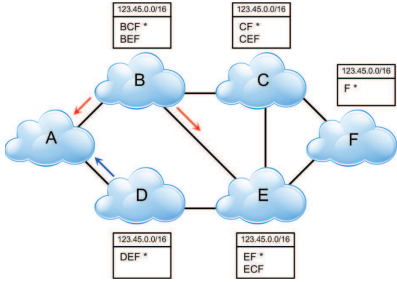
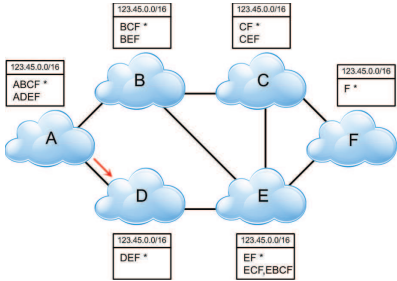
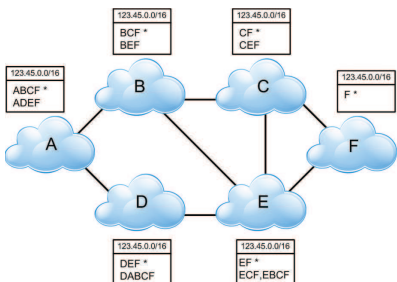
Each AS has a globally unique identifying number, called *Autonomous System Number* (ASN). ASNs are assigned and managed by the Internet Assigned Numbers Authority (IANA) [14]. ASNs are used to identify the path to a prefix as a sequence of ASes that the traffic will go through before reaching its destination. Accordingly, received advertisements are concatenated with local ASN, and propagated, AS by AS, to form the routing path. This path is used to forward the traffic to the destination prefix. Table 2.1 shows an example of how BGP works to advertise an IP prefix from one network to all the other networks.

### **2.1.6 BGP Security Issues**

BGP suffers from many security weaknesses [15]. There are many vulnerabilities in the design of BGP, as it was originally designed to communicate between trusted parties. However, the growth of the Internet changed that trust model, and security became a critical issue. There are many recent research efforts that propose solutions to the BGP security weaknesses. However, none of the solution is widely deployed in the Internet.

Some attacks on BGP only affect the neighbor router, some affect the complete neighbor AS, and others have larger scale that can affect the whole Internet. For example, *Session Termination* [15] attacks can close the BGP session between two routers when the attacker inserts a message that closes the connection. Another

Table 2.1: Example of how BGP propagates reachability information throughout the network

	<p>Network <math>F</math> starts by sending its IP prefix 123.45.0.0/16 to its neighbor ASes, <math>C</math> and <math>E</math>.</p>
	<p>Networks <math>C</math> and <math>E</math> now have a path to the prefix 123.45.0.0/16. Since this is the only path, it is selected as the best path. They propagate the newly learned path to all their neighbor ASes except to the source of that path, <math>F</math>.</p>
	<p>Network <math>C</math> has learned a new path to reach <math>F</math>, through the route <math>CEF</math>. Since there are more than one path, a best-path selection process is run, and the first path, <math>CF</math> is selected. Similarly, network <math>E</math> has learned the path <math>ECF</math>, but the selection process selects the first path, <math>EF</math>. Therefore, no changes have been made in <math>C</math> and <math>E</math>.</p> <p>Network <math>B</math> has learned two paths to <math>F</math>. It selects the best one, <math>BCF</math> for example, and propagates it to its neighbors. Network <math>D</math> also propagates the path it learned to its neighbors.</p>
	<p>Network <math>A</math> has learned two paths to 123.45.0.0/16, <math>ABCF</math> and <math>ADEF</math>. It selects the best path, <math>ABCF</math> in this case, and propagates it to its neighbor AS, <math>D</math>. It does not send that path to <math>B</math> because this is the source of the selected path.</p> <p>Network <math>E</math> has learned a third path, <math>EBCF</math>. It runs the best-path selection process, which selects <math>EF</math> again. Therefore, no changes are made.</p>
	<p>Network <math>D</math> receives the new path, <math>DABCF</math>. It then runs the selection process to select the best path, which results in selecting <math>DEF</math>. Hence, no more changes are made.</p> <p>The internetwork now is stable, and all networks have learned and selected the best paths to reach the prefix 123.45.0.0/16.</p>



attack, *Prefix Hijacking* [16], happens when a BGP router advertises an IP prefix originated from another AS, and claims that it is the originator. This attack can affect all of the Internet traffic that is destined to that prefix, as it will go to the attacker AS, which may *blackhole* all that traffic by dropping it.

One of the issues with BGP is the inability to control how traffic is routed through ASes. The received prefix reachability paths can only be considered as “promises”. There is no way to ensure that traffic will actually be routed through these paths. Practically, routers may provide signaling paths (the list of ASes that propagated the BGP update messages), which are not necessarily the same as the forwarding paths (the list of ASes traversed by data packets) [17]. Moreover, many networks use load-balancing and multihoming techniques to distribute traffic over multiple links. Thus, the traffic may go through different paths than the advertised ones, and may go through ASes that the traffic originator does not know about. The way BGP is designed allow the network to control only which neighbor AS will receive the packet, but not how that neighbor AS, or any other AS in the remainder of the path, will handle that packet.

Although this issue does not affect the delivery of traffic, as it will reach the destination on any path, it raises many security concerns. Packets may go through ASes that the traffic originator is unaware of, as they do not appear in the AS path. The presence of a malicious AS in any path to the destination, not necessarily the best path, results in the potential risk of routing the packets through that malicious AS.

A malicious AS may monitor, record, or even modify packets that are routed through it. It may also *blackhole* the traffic that belongs to a specific network, i.e., drop all the packets originated from or destined to the victim network. It can also deny providing routing services for that particular network, preventing it from accessing many destinations, namely the ones that are reachable through paths that go through the malicious AS.

## 2.2 Related Work

Before introducing the problem that this thesis addresses, it is important to address the related work done in the area of Internet resilience against different types of attacks.

Due to the growing importance of the Internet, much work has been done that studies the Internet resilience against different types of outages, failures, and attacks. Internet unavailability takes place due to either accidental or malicious causes. Hardware and/or software failures, misconfiguration, and traffic congestion are non-malicious activities that may cause Internet unavailability. Much work has been done to address these issues in the physical, routing, and application level [18–21].

Malicious activities that may cause Internet unavailability include denial of service (DoS) attacks, security breaches, terrorist attacks, intended hardware failures, and deliberate Internet denial by service providers. Most of the research that has been done in this area targets DoS attacks and security breaches [22–25]. Fewer

research efforts target terrorist attacks and intended hardware failures [26, 27].

The malicious act of Internet access denial by service providers has not been researched thoroughly. The Internet denial can take place at the physical, routing, or application layer. In this thesis, we introduce the problem of Internet denial at the routing level by malicious service providers, and propose a solution to tackle this problem.

## 2.3 Internet Denial by Malicious ISPs

### 2.3.1 What is Internet Denial

Most of the ASes that are close to the Internet core are owned by tier-1 and tier-2 ISPs. The networks of those higher-tier ISPs are large, as they cover multiple continents. Internet traffic, sent from a host on one network to a destination on a different network, is likely to go through multiple ASes. One or more of these ASes is a higher-tier ISP.

As shown in the discussion of the BGP security issues in section 2.1.6, traffic may go through networks and ISPs that are not announced in the BGP paths. It is not possible to determine the exact path of packets because each packet may travel on a different route to the destination. Response packets, sent back from the destination to the source, may also go through different paths than the paths used by the request packets.

We define *Internet denial* as the process of filtering transit traffic to drop

packets that belong a specific victim network. The ISP configures its network to drop, or *blackhole*, some or all the traffic that is originated from or destined to one or more IP prefixes. We assume that the ISP will use the network-layer information, namely the source IP address and the destination IP address, to determine if a packet belongs to the targeted network.

### 2.3.2 Identifying Packet Source and Destination

IP addresses are allocated as continues blocks of addresses. The *Internet Assigned Numbers Authority* (IANA) manages the IP address space allocations globally [28]. It cooperates with a number of *Regional Internet Registries* (RIRs) to assign IP address blocks to ISPs and other entities.

The mapping of IP addresses blocks to their owners is publicly available through the *WHOIS* protocol [29], a lookup protocol used to query information about specific Internet resources, such as IP address blocks and domain names. These databases are maintained by the RIRs, and can be used to determine information about the assignee of IP address blocks, such as the organization or ISP name and their country. Some services provide reverse lookup; they show all the IP address blocks that belong to the selected country [30].

This information can be used by malicious ISPs to determine which IP address blocks belong to the targeted network, organization or country, and perform Internet denial on these IP address blocks.

### 2.3.3 Motivations for Internet Denial

From a business point of view, an ISP that performs Internet denial on a network is risking its reputation. ISPs are supposed to provide the promised service of traffic routing without such filtering or denial. Hence, when the victim networks detect and report the act of Internet denial, that ISP may lose its reputation, and eventually its customers.

However, there are many other forces and motivations that may push an ISP to perform Internet denial on an organization or a country. An attacker that targets a specific organization can perform the attack at the ISP-level, by hacking into the ISP's network and reconfiguring it to block that targeted network.

Internet denial could also be driven by political motivations. Governments may enforce their ISPs to block some services from a specific country or region.

Many large services and networks have been attacked recently for political motivations. Gmail, for example, had many recent attacks targeting email accounts of Chinese human rights activists [31]. Twitter, a popular social network, has also been attacked recently by hackers from Iran [32]. These type of attacks are driven by political forces.

These reasons, and many others, can encourage ISPs to perform Internet denial on a specific network. The potential risk and impact of Internet denial could be critical. Therefore, solutions for this problem should be studied and deployed.

### 2.3.4 Impact of Internet denial

The impact of Internet denial depends on the location, size and connection topology of the malicious ISP. Lower-tier ISPs can only cause Internet denial if they exist in the route of the traffic. For example, a server that is located behind a malicious tier-3 ISP is not reachable by the victim network. Higher-tier ISPs, on the other hand, may cause larger impact. A malicious tier-1 ISP, for example, can block the victim network from accessing a large portion of the Internet and/or being accessed from the rest of the Internet, causing Internet isolation.

Because tier-3 ISPs do not act as transit for other networks, they only carry traffic that belongs to their networks. Therefore, a malicious tier-3 ISP can only block access to its own network. Hence, the impact of that ISP is only limited to a small set of hosts and services. Figure 2.4 shows a simplified model of the ISPs on the Internet. All other networks except the malicious one are accessible by the victim network, resulting in a very limited impact of Internet denial by a tier-3 ISP.

Malicious higher-tier ISPs can cause more impact as they can block not only traffic that belongs to their networks, but also all other traffic that passes through them as transit. A malicious tier-2 ISP, as shown in figure 2.5, blocks access to its own network, and to all its customer ISPs' networks. Moreover, it is also possible that ISP 6 in figure 2.5 becomes unreachable if the path to reach it goes through the malicious ISP. For example, if a packet is sent from the victim network to ISP 6, and it is sent through the path 3-1-2-4-6, ISP 4 will drop the packet, and ISP 6,

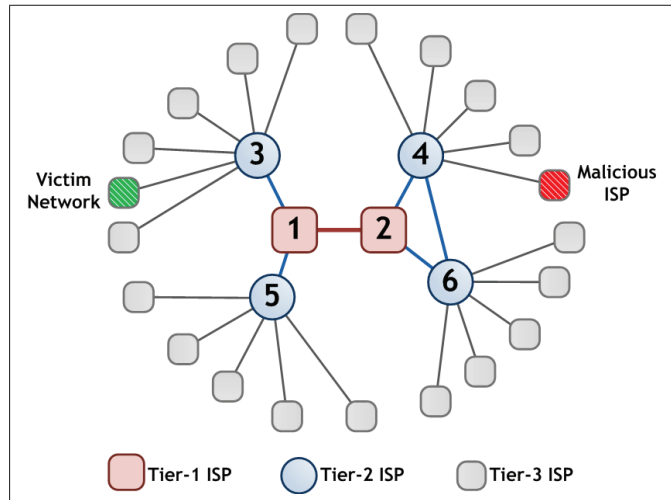


Figure 2.4: The impact of malicious tier-3 ISPs is limited to their networks and services

and all its customer networks, will become unreachable from the victim network.

Internet denial by tier-1 ISPs has the most critical impact. A malicious tier-1 can isolate the victim network and block it from accessing large portion of the Internet. An example is shown in figure 2.6, where all the networks for ISPs 2, 4, and 6, and their customer networks, are unreachable from the victim network.

Because of the critical impact that a malicious higher-tier ISPs can cause, solutions to this problem are needed. The next chapter describes a solution based on Network Address Translation (NAT).

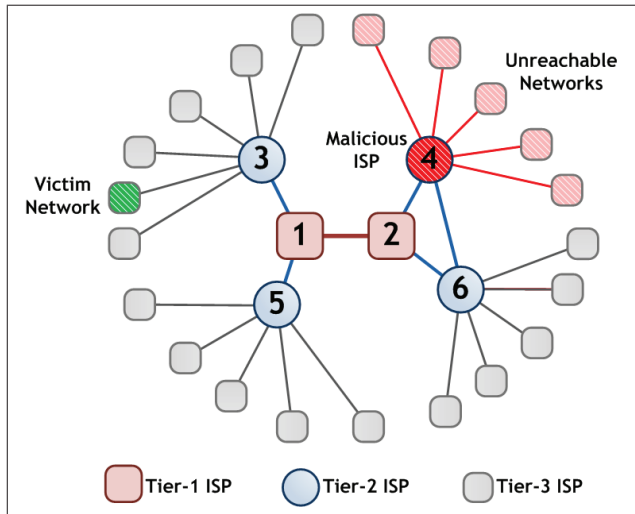


Figure 2.5: A malicious tier-2 ISP causes a large network to be unreachable

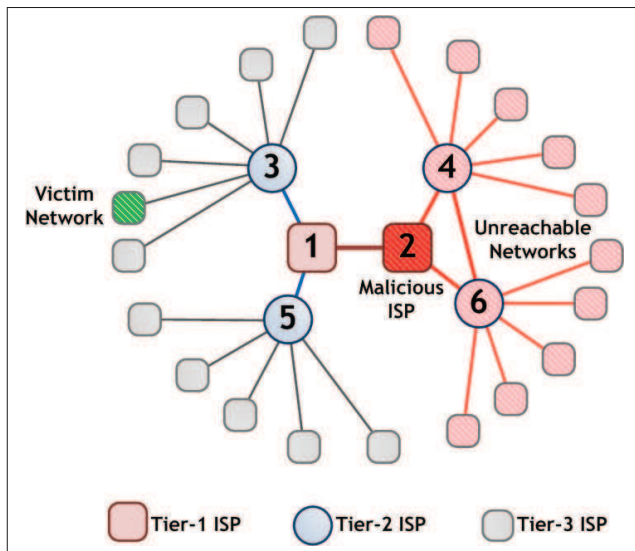


Figure 2.6: Malicious tier-1 ISP block a large number of networks causing Internet isolation



## CHAPTER 3

# NAT-BASED SOLUTION TO INTERNET DENIAL

Before looking into the proposed NAT-based solution to the Internet denial problem, we will take an overview of the different approaches to resolve this problem.

### 3.1 Solutions to Internet Denial

Internet denial by a malicious ISP happens when two conditions are met:

1. The traffic goes through the malicious ISP's network.
2. The malicious ISP drops packets that carry that targeted source or destination IP addresses.

Hence, the Internet denial problem can be resolved by eliminating one or both of these conditions. Two classes of solutions can be considered: solutions to

control the traffic path, so that it does not pass through the malicious ISP; and solutions to prevent traffic from being dropped at the malicious ISP by concealing the traffic identity.

### 3.1.1 Control the Traffic Path

The first class of solutions depends on preventing the traffic from being sent through the malicious ISP. Although BGP provides reachability information that includes the AS-path, it does not allow a network to control the actual path of its traffic. A network can only select which neighbor ASes will route its packets.

Controlling the outgoing and incoming traffic requires modifications or adjustments of the routing protocols. *Source Routing*, which allows the traffic originator to specify the path its traffic will travel through, is a solution to control the outgoing traffic so that it avoids the malicious ISP. However, the existing Internet protocols do not implement this type of routing. Modification of BGP is needed at all routers in the Internet to achieve this type of traffic control.

Quoitin et al. [33] proposed *BGP Tuning*, a set of techniques that controls incoming traffic. BGP tuning uses some techniques to influence the path selection process of remote ASes. Three techniques are presented: *AS-Path prepending*, where the length of the advertised AS-Paths is reduced to present it as a shorter path; *prefix splitting*, where the advertised IP prefix is disaggregated into a set of smaller IP prefixes to lead remote routers into selecting it as the longest prefix match; and the use of *Community*, where remote cooperating routers use the

community field in the BGP advertisements to identify the preferred paths.

*Virtual Peering*, also proposed by Quoitin [34], is a technique to control incoming traffic by using multi-hop BGP sessions. Remote ASes establish virtual peering tunnels to control the traffic destined to the local AS. This solution is not scalable as it requires all remote ASes to implement virtual peering and establish tunnels for all communications.

*Virtual Transit*, proposed by Alrefai [35], is a modification of virtual peering. The introduced difference is that remote ASes advertise the virtual-peering tunnel reachability information to their neighbor ASes, allowing them to use the same established tunnel to transmit traffic to the local AS. Virtual transit has better scalability than virtual peering, as only a portion of Internet ASes need to implement it.

### **3.1.2 Hide Traffic Identity**

The other class of Internet denial solutions is based on hiding traffic identity from the malicious ISP so that it does not identify the traffic's origin or destination. These techniques use IP addresses that are different from the blocked ones. Therefore, the malicious ISP will be misled into routing the traffic without filtering it.

#### **IP Address Replacement**

The first solution that comes to one's mind is to change the IP addresses of the blocked network into different ones. The victim network can just register an

IP block and use it instead of its current one. This solution may work for some time. However, the malicious ISP can easily detect that a new IP block is used by the blocked network, and will simply block it again. Hence, this solution is not robust.

## **Tunneling Protocols**

Network-layer encapsulation and tunnels are other methods of hiding the identity. Traffic is carried through a tunnel created between the two tunnel endpoints. Hence, packets are sent normally until they reach the first tunnel endpoint. Then, each packet is optionally encrypted then encapsulated as payload into another packet, then sent to the other tunnel endpoint. The intermediate routers will only see the two tunnel ends as the source and destination addresses. Packets then are decapsulated at the other end of the tunnel, and sent to their destination.

The simplest tunneling protocol is *IP-in-IP* [36], where the IP packet is encapsulated into another IP packet. Other tunneling protocols, such as *Internet Protocol Security (IPSec)* [37] and *Generic Routing Encapsulation (GRE)* [38], provide more security and encapsulation features, such as encryption and the encapsulation of different types of packets.

To implement tunneling as a solution to bypass Internet denial, at least two cooperating networks are needed as the endpoints of the tunnel. One of them should be located before the malicious network, and the other is located after it, so that the tunnel is established through the malicious ISP. The performance

degradation of using tunnels is relatively significant, because packet encapsulation adds an overhead to each packet, increasing the throughput requirements. Moreover, the use of encrypted tunneling protocols, such as IPsec, will add some computational overhead on the endpoints of the tunnel for the encryption and decryption operations. Although this solution is highly reliable once deployed, it does not work if no cooperating networks are found before and after the malicious ISP, such as the case of stub malicious networks. It also does not work when the destination host is within the malicious network.

### **Anonymous Routing**

One more technique of hiding the identity is the use of *Anonymous Routing* protocols. Anonymous routing provides means to hide the content of the packet, as well as the identities of the source and destination, from the routers that carry the traffic.

*Onion Routing*, first proposed by Syverson et al. [39], is one of the most popular anonymous routing protocols. The source host encrypts the message multiple times with different encryption public keys. Then it sends the encrypted message through a number of *onion routers* (i.e., network nodes that support onion routing). Each onion router decrypts one layer of the message, reads the routing information attached with the decrypted layer, and sends the message to the next onion router. When the packet reaches the last onion router, it decrypts the last layer, then sends the message to its destination. Intermediate onion routers are not aware of the content of the message, its original source or its final destination.

Other anonymous routing protocols, such as *Cashmere* [40], *Crowds* [41] and *Hordes* [42], are based on the same concept of Onion Routing, where messages are repeatedly encrypted initially and then decrypted layer-by-layer at the routers.

Anonymous routing protocols can be used as a solution for the Internet denial problem. They provide a reliable way of hiding the packet identity. However, the performance degradation of implementing such solution is very high [39, 43]. The solution would require a number of routers on the Internet that support the deployed anonymous routing protocol. In addition, a large cryptographic overhead is added to each router. Moreover, the number of hops that the message would traverse increases the end-to-end delay.

## **Network Address Translation**

*Network Address Translation* (NAT) is a technique that allows a large number of hosts to use a small set of IP addresses to communicate with other hosts on the Internet. A NAT router separates the network into two subnetworks, a private network, where the hosts are given private IP addresses; and the public network, where the NAT router is connected to the Internet by its public IP address.

NAT can be used as an identity hiding technique, by using a set of non-blocked IP addresses as the NAT's external IP addresses. All traffic will carry these non-blocked addresses when it is sent through the Internet.

The solution proposed in this thesis is NAT-based, where NAT is deployed at the gateway-level of the network to hide the identity of traffic by replacing its IP addresses. The next sections and chapters discuss this solution thoroughly,

including NAT design and deployment, scalability and performance evaluation, and solutions to the NAT consequent connectivity issues.

## 3.2 Network Address Translation

Internet was originally designed so that each entity, such as hosts and routers, has a globally unique IP address. The protocol that has been used is IP version 4 (IPv4), which uses a 32-bit address space, providing up to  $2^{32} = 4,294,967,296$  unique IP addresses. Later, it was clear that this address space is being exhausted at a faster rate that was not anticipated in the initial design of the protocol. Many technologies have been adopted as solutions to this problem. A newer version of the Internet Protocol, IPv6, was designed to be deployed instead of the current IPv4. IPv6 uses a 128-bit address space, which provides  $2^{128} \approx 3.410^{38}$  unique addresses. This address space is “large enough to have 155 billion IPv4 Internets on every square millimeter of the Earth’s surface, including the oceans” [44]. The deployment of IPv6 is, however, slow as IPv4 is still widely used nowadays. One of the reasons for this delay is the extensive use of a short-term solution, namely Network Address Translation (NAT).

### 3.2.1 What is NAT?

NAT is a technique that enables a number of hosts to use the same public IP address to connect to the Internet. It was first proposed by Paul Francis [45] as a temporary solution for the IPv4 address exhaustion problem. A typical

NAT network consists of a private network, where hosts are assigned private IP addresses, and the external, public network, through which the NAT router is connected using a public IP address. The NAT router and the private network behind it appear to the Internet as a single host, with a single public IP address. When a packet is sent from a host within the private network to the Internet, the NAT router *translates* the addresses on the packet header so that it replaces the private IP address with its public IP address. Similarly, response packets coming from the Internet to the private host are translated by replacing the destination public IP address with the host's private IP address.

Together with its main purpose of extending the IP address space, NAT also provides a level of security for the private network by hiding its internal addressing structure and topology.

### **IP Addressing in Private Networks**

The *Internet Assigned Numbers Authority* (IANA) has reserved three blocks of IP addresses for the addressing of entities in NAT private networks [46]:

- 10.0.0.0 - 10.255.255.255
- 172.16.0.0 - 172.31.255.255
- 192.168.0.0 - 192.168.255.255

All packets going through the NAT router must be translated, and the private IP addresses are replaced by the public IP address. Hence, packets on the public Internet should never carry private IP addresses.



### 3.2.2 Address Translation Process

NATs work on the basis of communication sessions, which are identified uniquely by the combination of: the source IP address and port number, and the destination IP address and port number. Together with the IP address translation, NAT also translates the source and destination ports for the transport-layer protocols, User Datagram Protocol (UDP) and Transmission Control Protocol (TCP).

When a host behind NAT sends a packet to the Internet, the NAT device intercepts the packet and replaces the source private IP address and port number by a public IP address and port number. Subsequently, it remembers this mapping by storing it in the *NAT Table*, and when an incoming packet is received with the same public IP address and port number, it replaces them with the private IP address and port number, and sends the packet into the private network.

#### Translation of Outgoing Requests

An example of how the NAT translation is done is shown in figure 3.1. The NAT router has the public IP address 3.3.4.4, and the private hosts have the IP addresses 10.0.0.1 and 10.0.0.2. The first host sends a packet destined to the server 8.8.9.9. The packet carries the source 10.0.0.1:543 and the destination 8.8.9.9:80 when it is sent from the host to the router. The NAT router then translates the source information to the public IP address and an external port, 3.3.4.4:3000. An entry is also added to the NAT table for future translations. This entry includes

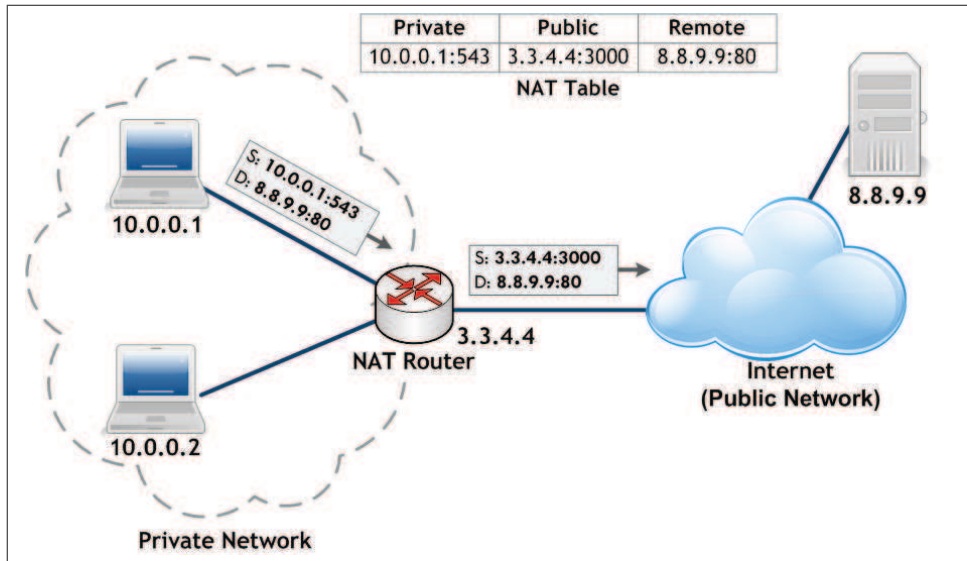


Figure 3.1: Outgoing packets sent through NAT are translated into public IP:port private, public, and destination IP addresses and ports.

When the server receives the request, the packet will appear as if it is coming from a host with the public IP address 3.3.4.4. Neither the server nor the routers that carried the packet to its destination are aware of the existence of a private network behind the NAT router that sent the packet. It only appears to them as if the NAT router is the actual host that sent that request.

Any subsequent packet that is sent from the same host to the same destination, or more specifically, from 10.0.0.1:543 to 8.8.9.9:80, is mapped to the same public IP address and external port, 3.3.4.4:3000. The NAT router checks the NAT table before translating the addresses to determine if this packet is part of an earlier session that was mapped to a specific external IP and port. If an entry is found, it uses the same information to map the packet, so that the destination receives it as if it is coming from the same host and port. Otherwise, if no entry is found,

the NAT router translates the packet to the public IP address and an available port, then it adds a new entry to the NAT table with the translation information.

### **Mapping of Incoming Responses**

When a response is sent from the server to the client behind NAT, it carries the router's public IP address and external port as the destination. When the NAT router receives the response packet, it will lookup a matching entry in the NAT table to determine which host in the private network is the correct destination of the packet. Once the NAT entry is found, the packet's destination is translated into the host's private IP address and port, then it is forwarded through the private network. The example in figure 3.2 shows the response sent from the server 8.8.9.9:80. The packet carries the destination 3.3.4.4:3000, which corresponds to the router's IP and port. The NAT router finds the entry in the NAT table and translates the destination of the packet to 10.0.0.1:543, then sends it to the private host.

The importance of the NAT table lies mainly in the mapping of subsequent traffic and responses. Without the NAT table, packets of the same stream might be mapped to different external ports, and the NAT router would not be able to properly handle any incoming packet.

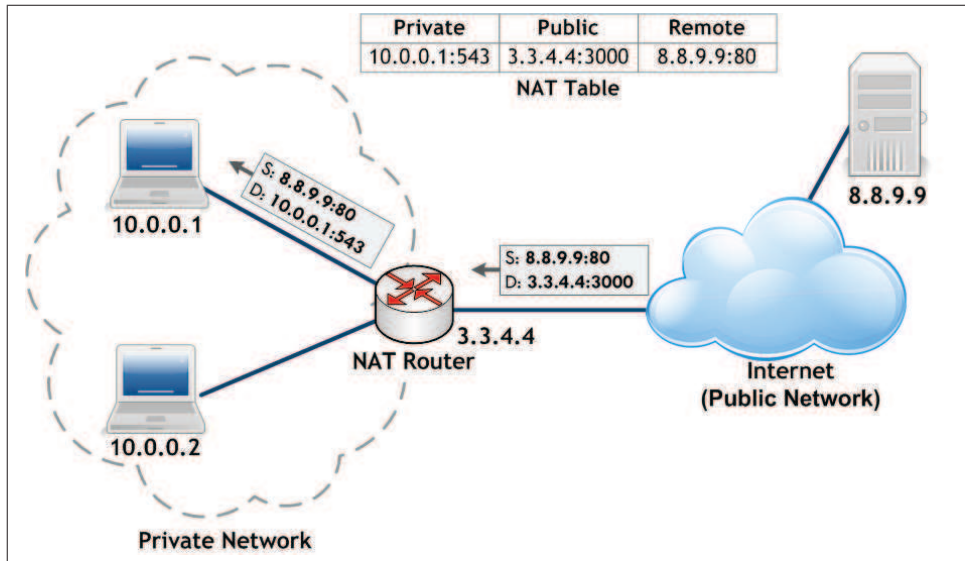


Figure 3.2: Incoming response packets received by NAT are translated into private IP:port

### 3.3 NAT as a Solution to Internet Denial

NAT hides a complete private network behind a single public IP address. The network appears to other entities on the Internet as a single host. Therefore, the structure, addressing, and topology of the private network is undetectable.

NAT can be used as an identity hiding technique to bypass Internet denial. The victim network uses NAT routers as gateways to connect to their ISPs, and uses a set of non-blocked IP addresses as the NAT's external public IP addresses. These addresses are not part of the IP ranges registered to the blocked network; they are obtained from a neighboring network. The outgoing packets, therefore, will not be blocked by the malicious ISP, as they will not be recognized as part of the blocked victim network.

### 3.3.1 Deploying Gateway-Level NAT

Implementing the NAT solution requires setting the gateway routers to use NAT to translate all traffic into the non-blocked public IP addresses. Once NAT is enabled and configured properly, clients within the victim network can send requests and receive responses. Even if traffic passes through the malicious ISP, it will not be recognized as traffic that belongs to blocked networks, and the malicious ISP will route it normally through its network.

### 3.3.2 Private Network Configuration

Although entities in the private network behind NAT are recommended to have IP addresses from the reserved private address blocks, they can still work with different IP address blocks if the NAT routers are configured properly. Therefore, for the NAT solution of Internet denial, entities within the victim network, including hosts and routers, do not need any modifications to adapt with the NAT solution. The only modification needed is at the gateway routers. NAT can be set in the existing gateway routers, or dedicated NAT routers can be used as a layer between the private network and the gateway routers.

In the typical NAT usage, hosts and routers are assigned private IP addresses from the reserved private IP blocks. However, in our solution, we will keep the existing IP addressing without changes. NAT routers can be set such that they recognize the internal IP address blocks as private addresses, and the translation is done between the internal IP blocks and the external public IP addresses. The

translation from one address space to another is also known as *IP masquerading*.

There are many advantages of keeping the same addresses. The NAT solution would be transparent to the clients as they do not have to make any changes in their networks. Moreover, local DNS servers do not have to update their records with private IP addresses, since no changes are made internally. In addition to that, keeping the same addresses would prevent addressing conflicts in case there are existing NAT networks within the victim network, an issue many NAT networks suffer from [47].

### **3.3.3 Local and Public DNS**

Hosts within the private network will be able to access services in the public Internet directly through NAT. The *Domain Name System* (DNS) lookups of public domain names are done normally through the DNS servers in the public network, as shown in figure 3.3. However, in order to lookup a domain name of a host within the private network, the lookup should be done through a local DNS server that is located in the private network, as shown in figure 3.4. This does not require any changes or additions to the network, as the authoritative DNS servers of local services are placed within the local network.

Therefore, existing DNS servers would be able to map services within the private network, as well as the public Internet. No modifications of the DNS servers are needed.

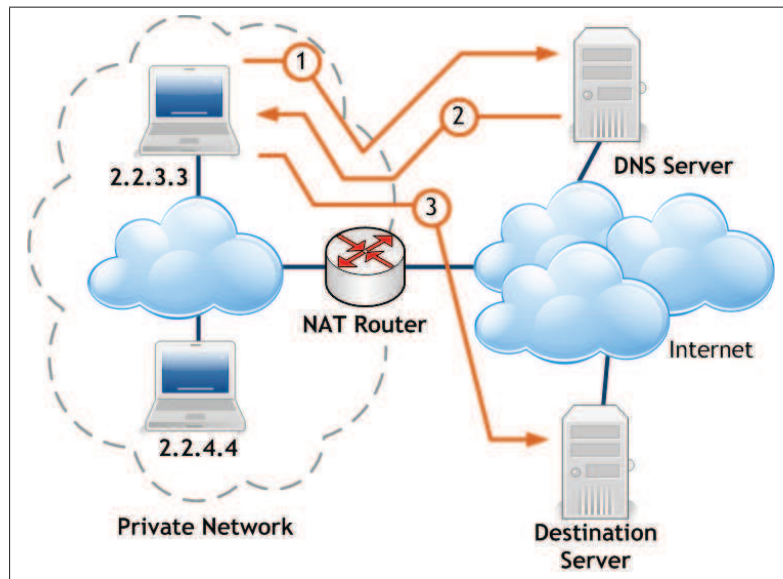


Figure 3.3: DNS lookup for external servers are done through DNS servers in the public Internet

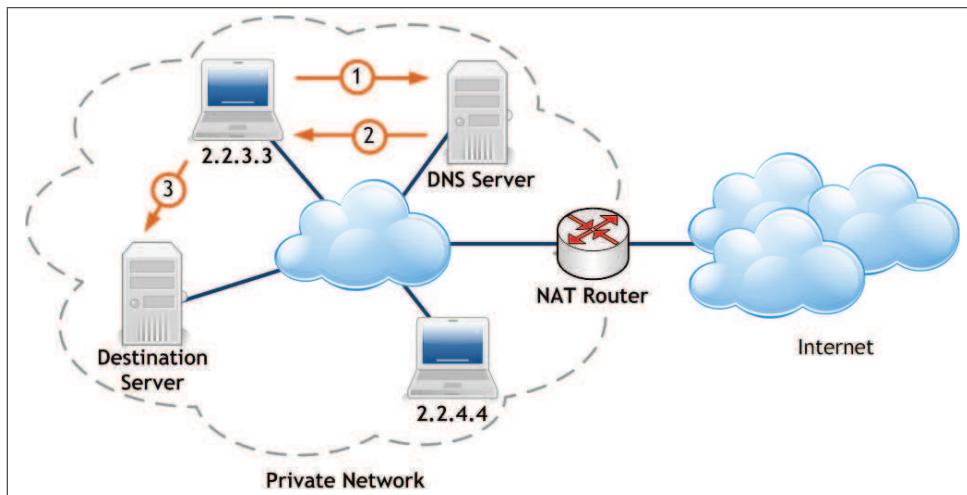


Figure 3.4: DNS lookup for internal servers are done through DNS servers located in the private network

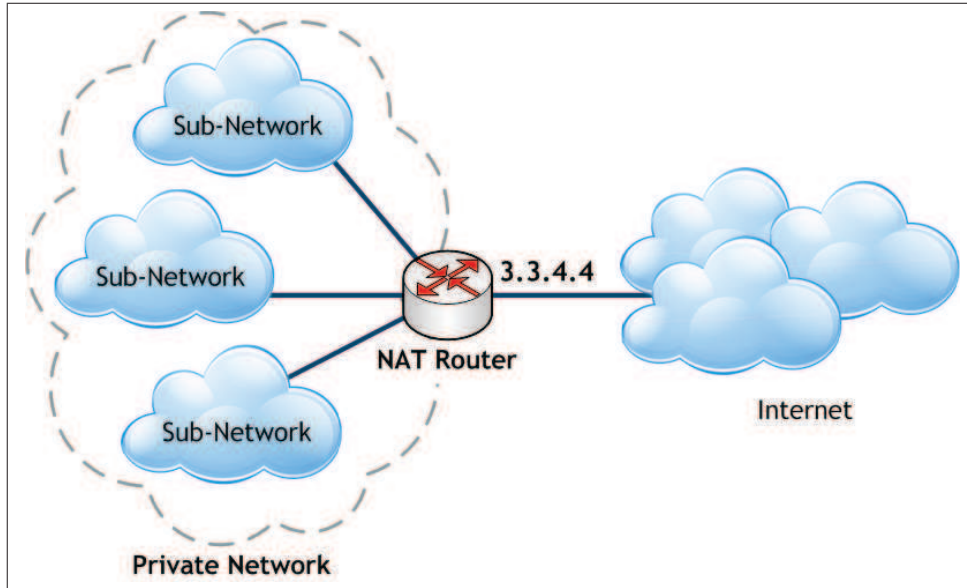


Figure 3.5: Initial design of the NAT solution, with a single NAT router mapping all traffic to a public IP address

### 3.3.4 Design Scalability

Because the proposed NAT solution is meant to solve the Internet denial problem, the victim network can range from a small LAN to an entire country. Therefore, scalability issues and limitations should be investigated.

Initially, we assume the use of a single router, as shown in figure 3.5. The NAT router is used to connect to the Internet, and all the traffic is translated into the public IP address 3.3.4.4.

#### Extending Mapping Space

The first scalability issue is the limited number of possible mappings. NAT maps each session to a single external port. The tuple of source IP:Port and destination IP:Port is used to map subsequent traffic to the same external port. TCP



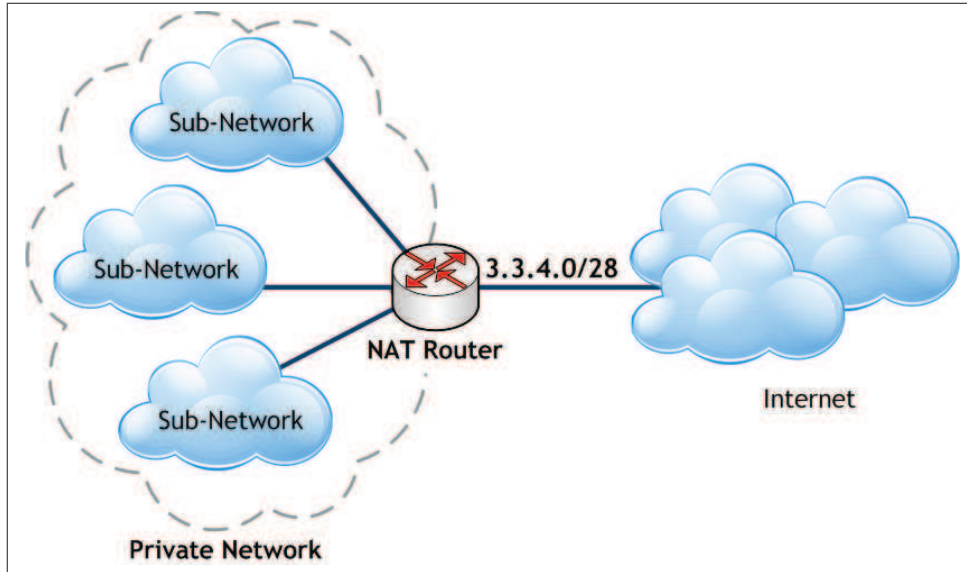


Figure 3.6: Extended NAT solution design using a pool of public IP addresses

and UDP use 16-bit port numbers, providing 65,536 ports. Ports from 1 to 1023 are called the “well-known ports”, as they are reserved for specific applications by the *Internet Assigned Numbers Authority* (IANA), and they should not be used as source ports. That leaves 64,512 ports that are usable as source ports. Hence, a NAT router can map up to 64,512 sessions for each public IP address at the same time. If there are more connections coming to the router, it may not be able to serve them as there are no more available ports.

This issue can be resolved by using a pool of public IP addresses instead of using a single public IP address. Adding public IP addresses increases the available ports exponentially, since every added address provides the complete port space to be used for mapping. Extending the initial design example, figure 3.6 shows the extended network where the NAT router is now using the IP pool 3.3.4.0/28, which consists of 16 public IP addresses, from 3.3.4.0 to 3.3.4.15.

## Load Balancing

Other NAT scalability issues include memory, bandwidth, and processing requirements. For each NAT mapping, an entry is added to the NAT table. Since a router can map up to 64,512 sessions at the same time, that much NAT entries are expected to be in the NAT table.

A NAT table entry requires about 160 bytes [48]. Therefore, a fully-utilized NAT table with 64,512 entries would require a little less than 10 megabytes of memory, which is much less than available memory in routers nowadays. Hence, the growth of the NAT table is not an issue when a single public IP address is used.

However, the use of pools of public IP addresses will significantly increase the required memory. For example, the NAT table resulting from the full mapping of a pool of 16 IP addresses would require 160 megabytes, which is considerably high. Therefore, router memory might become a limitation on the design.

Moreover, the NAT router has a limited processor power such that it might not be able to handle that much traffic. Bandwidth and processor limitations need to be considered as well.

To resolve these issues, load-balancing can be used by adding more NAT routers at the gateway level. Each NAT router handles a portion of the private network, and has its own pool of IP addresses. This method provides large scalability of the solution since more NAT routers can be added as needed.

The partitioning of the internal network can be done by the physical topology.

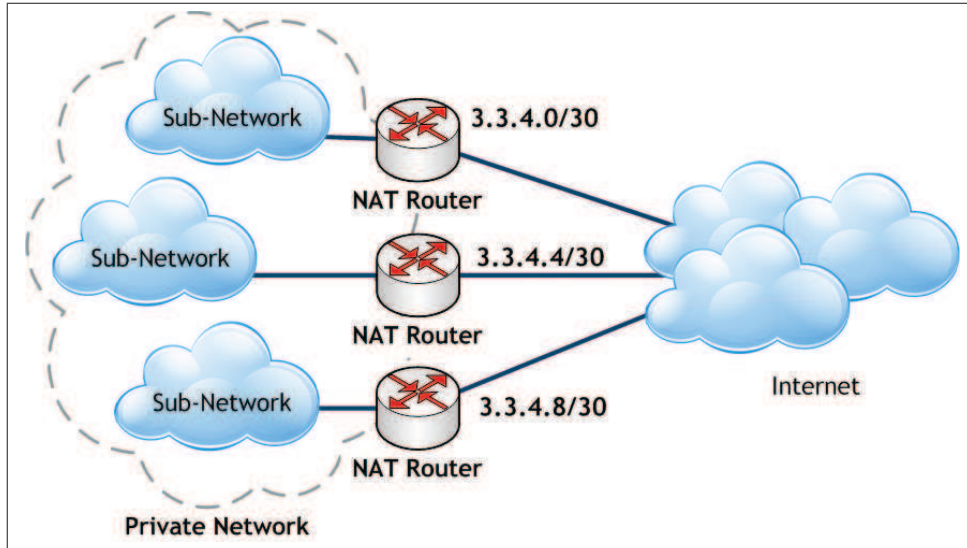


Figure 3.7: Extended NAT design using load-balancing over a number of NAT routers

The private network is partitioned into a number of subnetworks, and each sub-network uses its own NAT router to translate traffic. For example, if this solution is to be implemented on a country-level, the country's network can be partitioned by ISPs. Each ISP is a subnetwork that is connected to the International ISP using one or more NAT routers, as shown in figure 3.7.

### 3.4 Performance Evaluation

NAT adds some extra operations that take place in the NAT routers, which may effect the performance. We will evaluate the performance degradation, if any, of deploying the NAT solution.

### 3.4.1 NAT Processing Delay

#### Extra Processing Added By NAT

Enabling NAT in a router introduces a computational overhead that, theoretically, affects performance. NAT performs a number of added operations on packets. For each incoming packet, the NAT router changes the destination IP address and port. Similarly, for each outgoing packet, the NAT router changes the source IP address and port. The router also performs NAT table lookup to find a matching entry, and if none is found, it adds a new entry. TCP packets have a packet-checksum in their TCP header, which also needs to be recomputed.

However, many router vendors, such as CISCO, suggest that the extra delay added by enabling NAT is very small and negligible [49] because routers are designed to minimize the NAT computational overhead. NAT may even have *zero* impact on performance, as some routers, such as Juniper's SSG500 [50], are designed using a session-based architecture where the router keeps track of complete connection sessions and is aware of the packet's transport-layer information.

Most popular network simulators, such as OPNET [51] and ns-2 [52], do not consider NAT processing delay in their simulations [53]. In order to correctly evaluate the performance of NAT, a correct delay model of NAT needs to be implemented in the simulator.

## Related Work on Packet Processing Delay

Clark et al. [54] have studied the overhead of the Transmission Control Protocol (TCP). They measured the computational overhead done at the transport layer, such as TCP checksum computation, and memory read and write accesses. They concluded that the TCP overhead is very small, and it is not the source of processing overhead. The overall overhead per packet does not exceed a fraction of a millisecond.

That study was done in 1989. Network processors have significantly been enhanced over the last two decades, and the measured TCP overhead would be even smaller by now. NAT computational overhead is somehow similar to the TCP overhead, as both are in the transport-layer, and they have similar computations, such as the checksum calculation. Hence, it is possible to approximate the NAT delay to the measured TCP overhead.

Ramaswamy et al. [53] have studied the network processing delay that packets experience. They estimate that on a 1Gbps network, the processing delay of complex packet modifications, including NAT, firewall, and IPsec encryption, is  $1,000\mu\text{s}$ , as shown in table 3.1. They model a simplified network processor to measure the end-to-end delay that a single packet experiences. They did not consider the effect on the overall throughput as routers are designed to improve performance by processing many packets in parallel using multi-core processors, and the processing overhead would have a significant effect only on the end-to-end delay of a single packet.

Table 3.1: Network delay components, showing the processing delays as estimated by Ramaswamy et al. [53]

<b>Delay</b>	<b>Simple Packet Forwarding</b>	<b>Complex Payload Modifications</b>
Transmission delay	10 $\mu$ s	10 $\mu$ s
Propagation delay	1,000 $\mu$ s	1,000 $\mu$ s
Processing delay	10 $\mu$ s	1,000 $\mu$ s
Queuing delay	0 ... $\infty$	0 ... $\infty$
Fraction of processing delay to total delay	1%	50%

### Negligibility of NAT Delay

Although the study performed by Ramaswamy [53] shows that processing delay is not very small, we still can consider it negligible for the NAT-based Internet denial solution for three reasons:

1. Ramaswamy's measurements are set on a 1Gbps LAN, where a delay in the order of microseconds is considered significant. In our solution, however, the Internet delay, which ranges from tens to hundreds of milliseconds, is much higher than the added processing delay. In the worst case, the added one or two milliseconds has no much effect on the overall performance.
2. Network processors nowadays are designed to provide high level of parallelism, using multi-core and pipelining technologies. As indicated by Ramaswamy, the added delay would only affect a packet's end-to-end delay. A flow of packets would not suffer from that much delay, since multiple packets will be processed in parallel.
3. The measured processing delay includes the sum of many operations: NAT,

firewall, and IPsec. The computations that NAT requires are much smaller than the more complex computations performed in IPsec encryption. Hence, only a small portion of the measured  $1000\mu s$  is due to NAT.

Therefore, the NAT processing delay is not expected to have any significant impact on the performance of the network, as long as the same network resources are available. This will be shown by performing simulations of NAT to evaluate its performance impact.

### **3.4.2 Simulation of the NAT Solution**

In order to evaluate the impact of implementing the proposed NAT solution on the network, simulations are performed. The OPNET [51] network simulator is used to perform these simulations.

The objective of the simulations is to compare the network performance before and after implementing the NAT solution. The used performance metrics are the end-to-end delay, the traffic throughput, and the packet drop rate. Different applications are tested under different traffic loads.

#### **Modeling of NAT Delay**

NAT delay is added to the packets whenever NAT is enabled on the local routers. The processing delay in OPNET is modeled in a simplified way. The delay suffered by each packet is the reciprocal of the forwarding rate of the router. For example, if a router model in OPNET has a forwarding rate of 100,000 packets

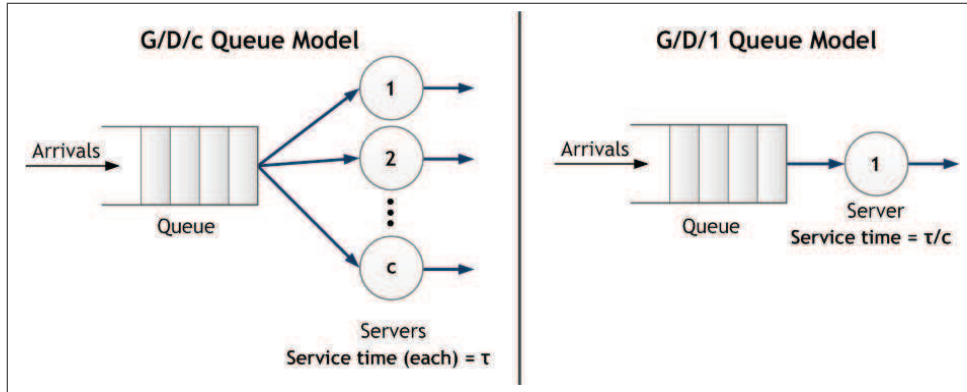


Figure 3.8: G/D/ $c$  and G/D/1 queuing models.

per second, the processing delay of each packet would be  $1/100,000 = 10\mu\text{s}$ . The single-packet processing delay of a real router may be higher, though. The OPNET simplified model does not take parallel processing into account.

The network processor in a router can process multiple packets in parallel. Some processors, such as Intel IXP2800 network processor [55], can process up to 128 packets in parallel. Hence, a router can be modeled as a G/D/ $c$  queuing system (arbitrary arrival rate, deterministic service rate), with a single queue and  $c$  servers, as shown in the left side of figure 3.8. The number of servers,  $c$ , represents the number of packets that the router can process in parallel. Assuming the service time for each server is  $\tau$ , OPNET simplifies this model by using a G/D/1 queuing model, with a single server that can process packets  $c$  times faster, as shown in the right side of figure 3.8. Hence, the service time is  $\tau/c$ .

This model simplification is not correct for measuring the delay of a single packet. The packet in the actual router will suffer a processing delay of  $\tau$ , regardless of the parallelism that the router supports. The two models are equivalent



only under the condition that the queue is not empty. In this case, all servers in the G/D/c model are busy processing packets. Therefore, the throughput is  $c$  packets per  $\tau$ , which is similar to the throughput of the G/D/1. This means that the simulation of processing delay in OPNET is correct only under the condition that there is a sufficient amount of traffic to keep servers busy. Therefore, the traffic bit rate used in the simulations should not be very low.

The delay is added to the service time of the single-server queuing system. Therefore, the simulated NAT delay must be  $1/c$  of the real NAT delay, in order to take parallelism into consideration. For example, to simulate a NAT delay of  $500\mu\text{s}$  (0.5 ms) on a router that can process 32 parallel threads, a delay of  $500/32 = 15.6\mu\text{s}$  is added to the simulation.

Based on Ramaswamy's work [53], a packet processing delay of  $1000\mu\text{s}$  is estimated on a router that runs NAT, firewall, and IPsec encryption protocol. Because NAT is less complex than the operations like firewall and IPsec, its delay is represented only by a small portion of the total processing delay. In the worst case, we select half of that processing delay, i.e.,  $500\mu\text{s}$ . For a router that processes 16 packets in parallel, the modeled delay should be  $500/16 = 31.25\mu\text{s}$ . This delay represents the worst case scenario when NAT is used. Therefore, real routers will have better performance and less overhead than the simulated one.

The range of simulated NAT delay values is between  $10\mu\text{s}$  and  $250\mu\text{s}$ . In reality, the range for real routers is between  $10\mu\text{s}$  and  $50\mu\text{s}$ . The remaining range, i.e., from  $50\mu\text{s}$  to  $250\mu\text{s}$ , does not reflect the real routers' performance. It is simulated

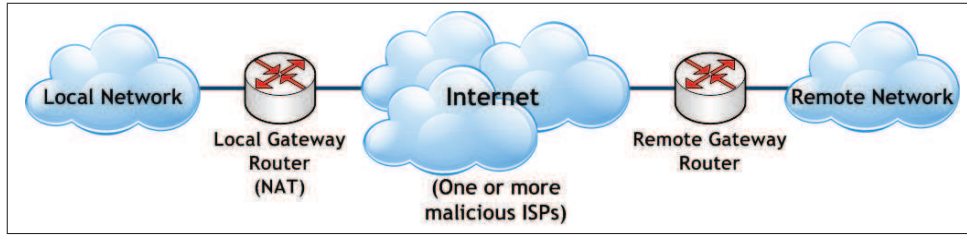


Figure 3.9: Simulated scenario to measure the effect of NAT delay on network performance

only to see the effect of high processing delay on performance.

### Simulated Scenario

The simulated scenario is shown in figure 3.9. It consists of two networks, local and remote. Each network consists of a Local Area Network (LAN) and a gateway router. NAT is enabled in the local network’s gateway router. An IP cloud, representing the Internet, is connecting the two gateway routers.

### Simulation Setup and Parameters

The local and remote networks are 100BaseT *Fast Ethernet* networks. Each network has a number of connected hosts that will serve as clients and servers for each application. The number of hosts on each network is set to 10. The gateway routers are based on the generic router model in OPNET. It supports many protocols, including BGP and NAT. Both routers are connected to the central Internet cloud using DS-1 links, which provide a data rate of 1.544 Mbps.

Two applications are simulated: FTP, which runs over TCP; and Video Conferencing, which runs over UDP. Each application is simulated under three traffic

scenarios: low, medium, and high traffic. The *low traffic* scenario uses about 25% of the available link's bandwidth, which is about 380 kbps. The *medium traffic* uses 50% of the bandwidth (about 770 kbps). The *high traffic* utilizes about 75% of the bandwidth (about 1,200 kbps). These scenarios are selected to evaluate the performance of NAT under different traffic loads.

Each simulation is run 5 times, and the average of the 5 results is taken. Performance is evaluated over three measurements: end-to-end delay, traffic throughput, and packet drop rate.

### **Simulation of End-to-end Delay**

The three scenarios of traffic are simulated for both UDP (video conferencing) and TCP (file transfer) to measure the end-to-end delay. End-to-end delay refers to the amount of time that a packet takes to travel from the client to the server, going through the local NAT router, the Internet cloud, and the remote gateway router; and including the transmission times, the queuing delays, and the added NAT delay.

The effect of NAT delay on the total end-to-end delay for UDP traffic can be seen in figures 3.10, 3.11, and 3.12. The figure shows the end-to-end delay for low, medium and high traffic; with and without NAT, versus the simulated NAT delay. When NAT is not enabled, the NAT delay is not taken into consideration. Hence, the end-to-end delay is constant for the NAT-disabled case. However, when NAT is enabled, the delay packets suffer to reach the destination increases linearly.

The added NAT delay is suffered by every packet that passes through the

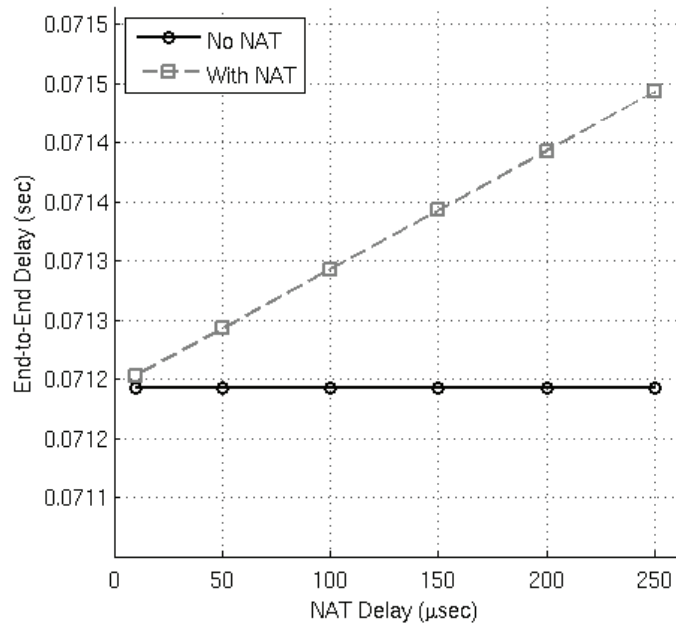


Figure 3.10: End-to-end delay for low UDP traffic

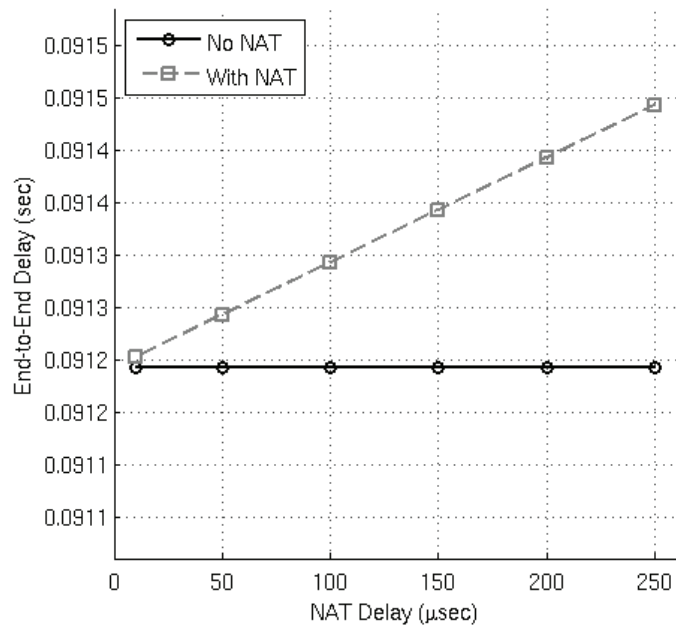


Figure 3.11: End-to-end delay for medium UDP traffic

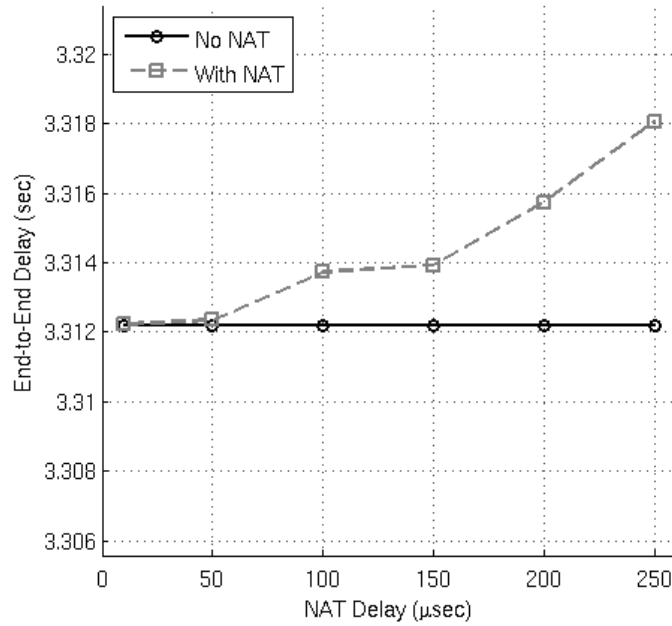


Figure 3.12: End-to-end delay for high UDP traffic

router. If we model the router as a G/D/1 queuing system, the delay suffered by an arriving packet is  $N\tau$ , where  $N$  is the number of packets in the system, and  $\tau$  is the processing time. An added NAT delay of  $\Delta\tau$  will result in increasing the processing time to  $N(\tau + \Delta\tau) = N\tau + N\Delta\tau$ . Hence, the increase of  $\Delta\tau$  causes a linear increase of the processing time by  $N\Delta\tau$ .

The increase of the end-to-end delay for UDP traffic is shown in figure 3.13. The increase is computed as  $(Delay_{NAT} - Delay_{NoNAT})$ . It represents the amount of delay increase that is caused by the introduction of NAT. It is clearly noticed that the increase of traffic causes large increase in the end-to-end delay. NAT delay below  $100\mu s$  have very negligible impact on the end-to-end delay. However, as the NAT delay increases, its impact becomes more significant for higher traffic.

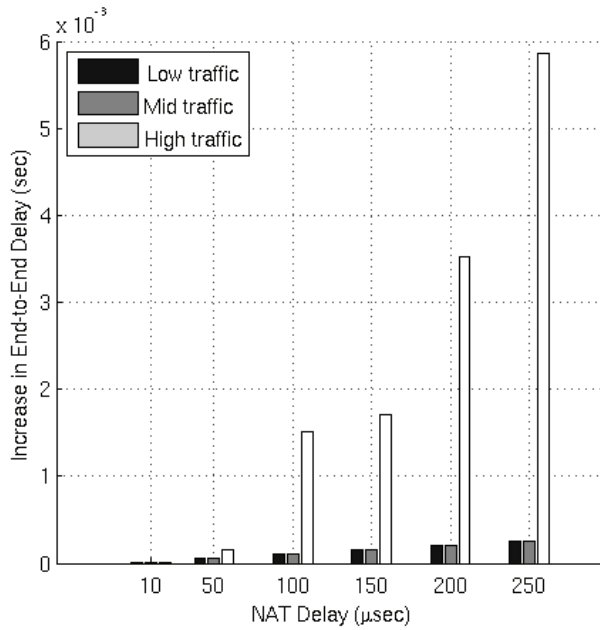


Figure 3.13: Increase of end-to-end delay for UDP traffic

Figure 3.14 shows the relative increase with respect to the original end-to-end delay. The relative increase is computed as  $\frac{Delay_{NAT} - Delay_{NoNAT}}{Delay_{NoNAT}}$ . We can see that for small NAT delays, specifically below  $100\mu s$ , the effect of NAT does not exceed 0.1% of the total end-to-end delay. Larger values of the NAT delay cause a relatively higher increase in the end-to-end delay. However, the maximum delay in the highest NAT delay still does not exceed 0.4% of the total delay. We also notice that the relative effect of NAT delay is lower when the traffic is high. This is because higher traffic results in higher queuing delay, which eventually becomes much more significant than the NAT delay. Hence, the relative effect of NAT delay is lower.

The other application, file transfer over TCP, has a similar behavior of the end-to-end delay. We can see in figures 3.15, 3.16, and 3.17 that the end-to-

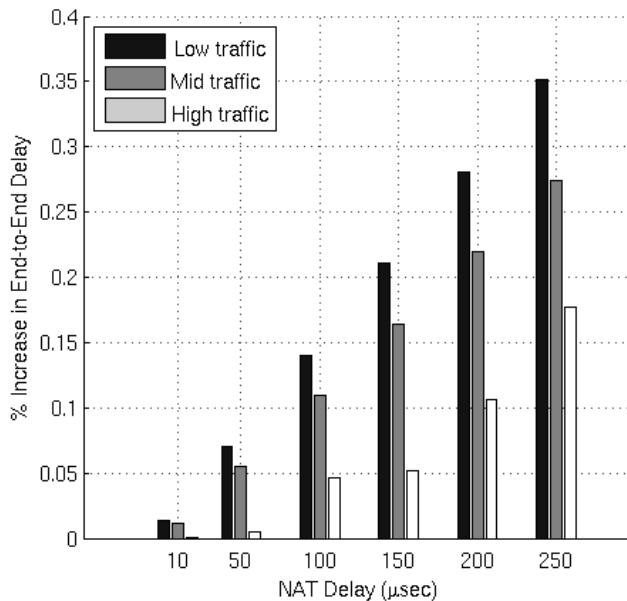


Figure 3.14: Relative increase of end-to-end delay for UDP traffic

end delay increases linearly with the increase of the simulated NAT delay. It is noticed that the curves are not as smooth as the ones in the UDP simulations. The reason is that the simulated application, FTP, does not use constant bit rate as the video conferencing does. FTP is simulated as requests for file downloads or uploads. Moreover, FTP runs over TCP, which requires the overhead of connection establishment. In addition, TCP packets have larger header than UDP ones. These factors increase the variance of TCP traffic in the simulation. The overall trend of the end-to-end delay, however, is still linearly proportional to the NAT delay.

Figure 3.18 shows the amount of increase caused by NAT. Again, lower NAT delays have insignificant impact on the end-to-end delay, whereas higher NAT delays have small effect. The relative increase of end-to-end delay, shown in

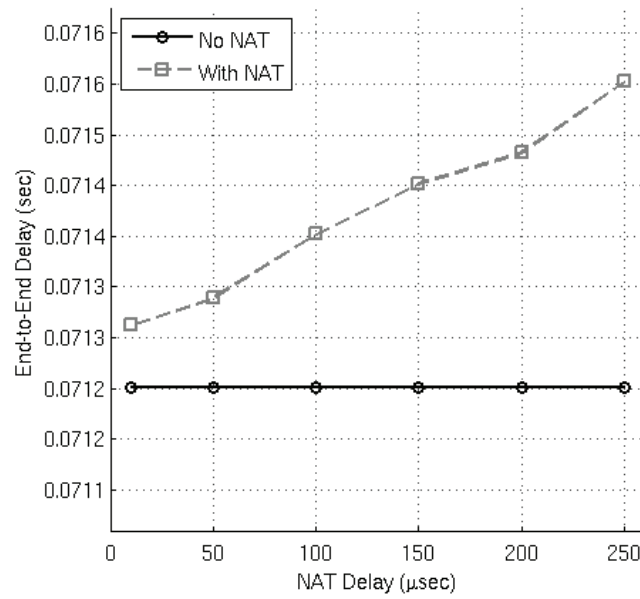


Figure 3.15: End-to-end delay for low TCP traffic

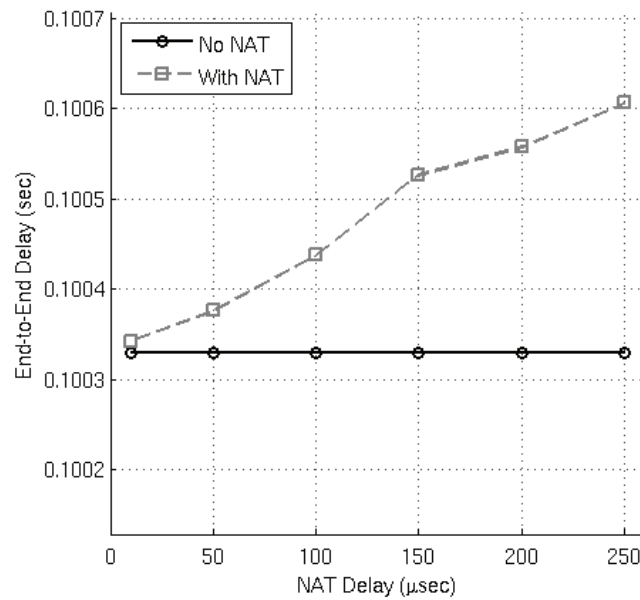


Figure 3.16: End-to-end delay for medium TCP traffic



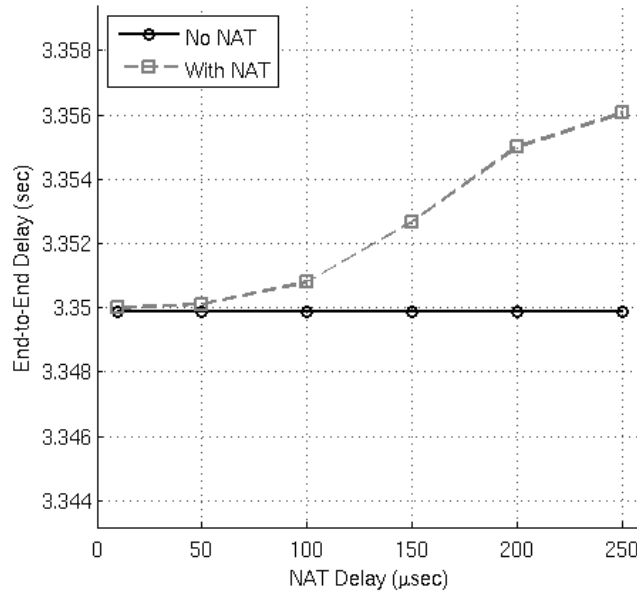


Figure 3.17: End-to-end delay for high TCP traffic

figure 3.19, is higher than the increase of UDP traffic. The above mentioned variance factors cause this increase. However, the highest relative increase does not exceed 0.45% of the end-to-end delay, and that increase occurs only when the NAT delay is more than  $200\mu\text{s}$ , which is an extremely high delay that NAT routers do not really reach.

It can be concluded that NAT does not have any significant impact on the end-to-end delay. The maximum increase of the end-to-end delay does not exceed 0.5% in the worst case when the NAT delay is higher than  $200\mu\text{s}$ , which is an extremely unrealistic scenario. For a router that, for example, can process 16 parallel packets, the modeled  $200\mu\text{s}$  delay would represent a delay of 3.2ms on the router. Such delay is very high, and all routers are much faster than that. Therefore, in the reasonable range for the modeled NAT delay, which is between

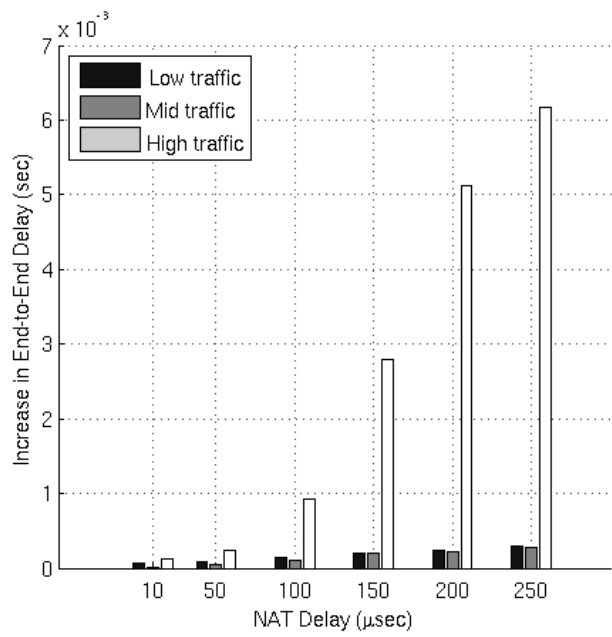


Figure 3.18: Increase of end-to-end delay for TCP traffic

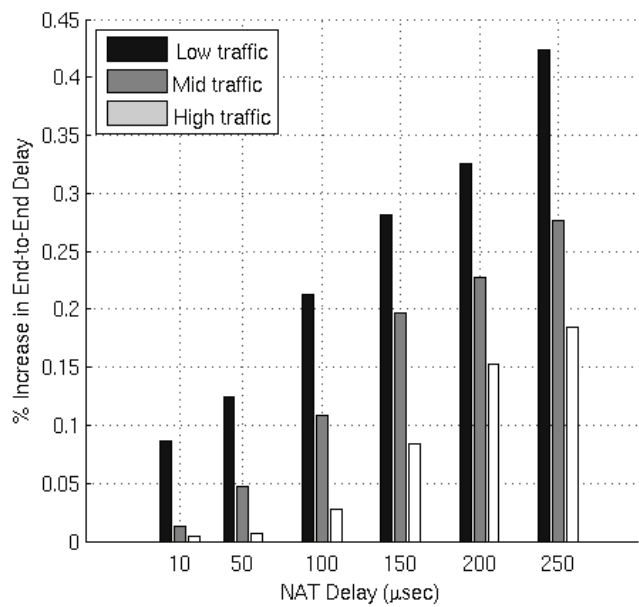


Figure 3.19: Relative increase of end-to-end delay for TCP traffic

10 and 50 $\mu$ s, NAT adds very small and negligible effect on the end-to-end delay.

## Simulation of Traffic Throughput

The throughput is measured throughout the simulations in order to see the impact of NAT on the amount of transmitted and received traffic. The same simulation setup is used, where the three scenarios of 25%, 50% and 75% traffic load are simulated, and the NAT delay is varied between 10 and 250 $\mu$ s.

Throughput is measured as the amount of application traffic sent and received by the hosts per second. The simulation is set to measure the throughput at the client side.

For the cases of low and medium traffic, NAT does not have any effect on the throughput; both scenarios, with and without NAT, have exactly the same measured throughput. In the case of high traffic, NAT only starts to affect the throughput when the NAT delay is very high, i.e., more than 150 $\mu$ s. Figures 3.20 and 3.21 show the throughput of high UDP and TCP traffic, respectively. The degradation of throughput is due to the high NAT delay which slows down the processing of packets, and causes the router queue to be filled with waiting packets.

The relative decrease of throughput, which is computed as  $\frac{(Throughput_{NoNAT} - Throughput_{NAT})}{Throughput_{NoNAT}}$ , is shown in figure 3.22. It can be noticed that the degradation of throughput starts earlier in TCP traffic, as a NAT delay of 150 $\mu$ s causes a small decrease in the throughput. The maximum relative decrease is less than 0.3% of the total throughput, which is insignificant. Nevertheless, in the realistic NAT delay range, the throughput is not affected at

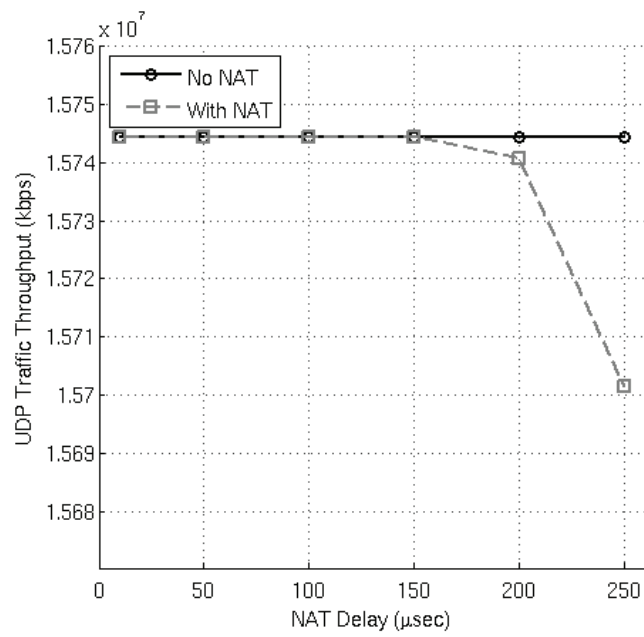


Figure 3.20: Throughput of high UDP traffic

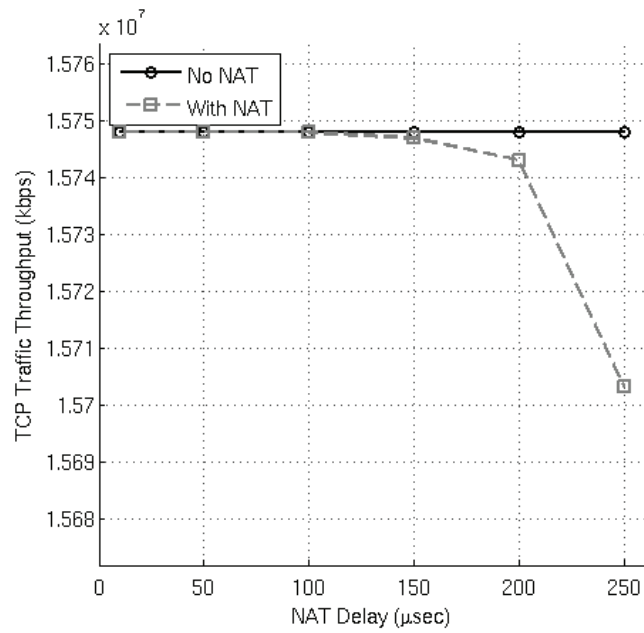


Figure 3.21: Throughput of high TCP traffic

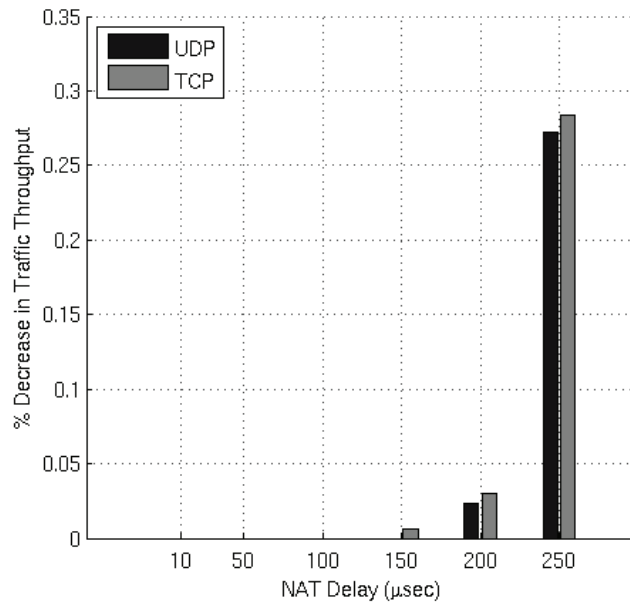


Figure 3.22: Relative decrease of throughput for TCP and UDP traffic

all. We can conclude that NAT does not affect the throughput of the network except at the extreme cases of high NAT delay, and even in this case, the performance degradation is negligibly small.

### Simulation of Packet Drop Rate

The reason of the throughput degradation is the large amount of NAT delay that causes queuing of packets. New packets are dropped when the router queue is full. The packet drop scenario is simulated in order to study the effect of NAT on the amount of dropped traffic.

Dropped traffic is measured at the NAT router, where packets are actually suffering the NAT and queuing delays. No packets are dropped in the cases of low and medium traffic. High traffic, on the other hand, can cause traffic dropping,

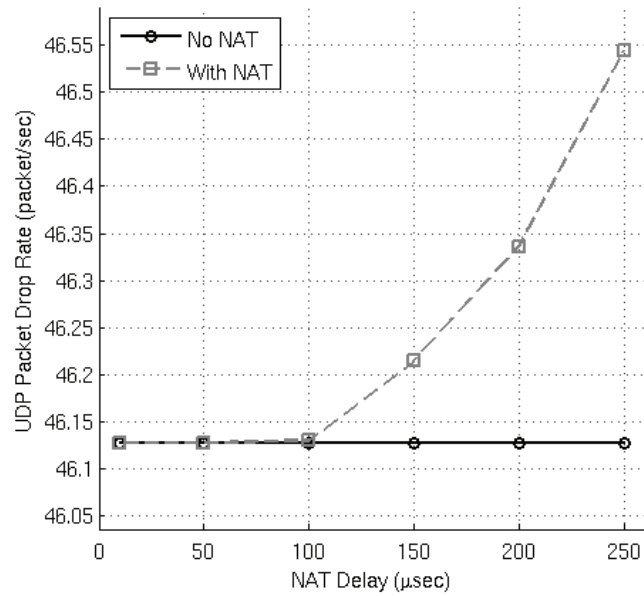


Figure 3.23: Packet drop rate for high UDP traffic

but only with high NAT delays. Figures ?? and 3.24 show the amount of dropped packets for high UDP and TCP traffic. Low NAT delays do not affect the number of dropped packets at all. However, the increase of the NAT delay beyond  $100\mu\text{s}$  results in an increase in the packet drop rate. It is also noticed that the TCP drop rate is higher than that of UDP due to the overhead of TCP.

Figure 3.25 shows the relative increase of packet drop rate, computed as  $\frac{(\text{DropRate}_{NAT} - \text{DropRate}_{NoNAT})}{\text{DropRate}_{NoNAT}}$ . We see that NAT increases the packet drop rate only when the modeled NAT delay is  $150\mu\text{s}$  or more. The effect of NAT in this case does not exceed 1.3% of the total dropped packets for TCP, and is less than 1% for UDP. Such increase is considered small, and hence, the NAT effect on the packet drop rate can also be considered negligible.

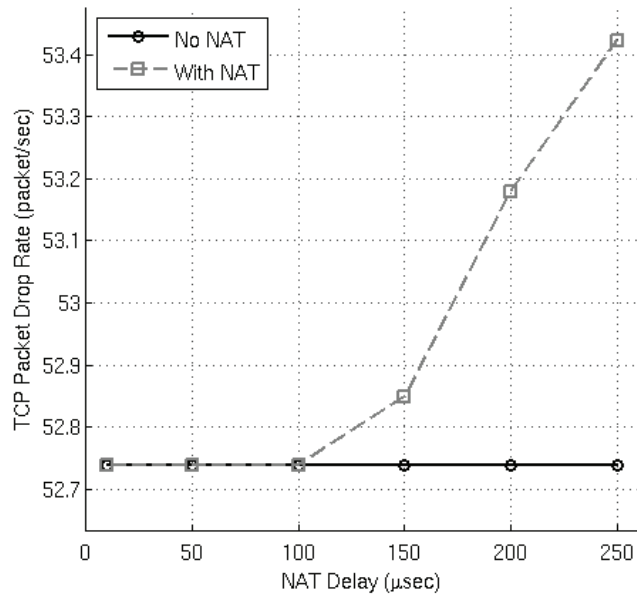


Figure 3.24: Packet drop rate for high TCP traffic

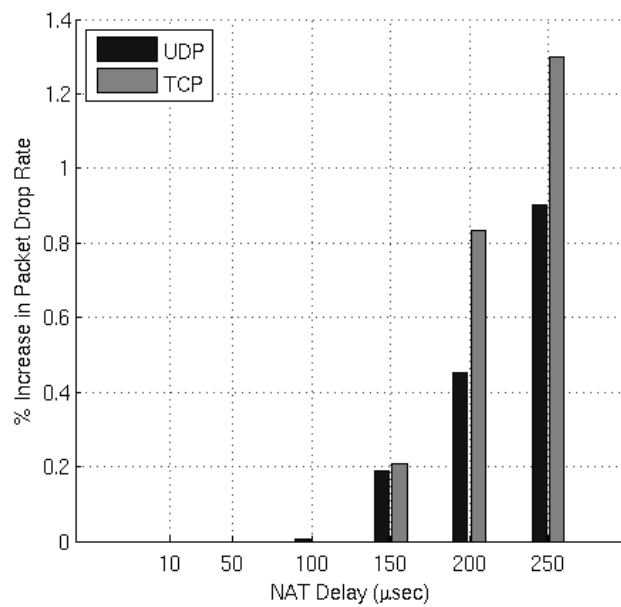


Figure 3.25: Relative increase of packet drops for TCP and UDP traffic

## 3.5 Conclusion

It was shown that NAT does not have any significant impact on the performance of the network. The performance degradation that was measured in simulations happens only when the NAT delay is set to a very large value. Therefore, most existing NAT routers can perform the NAT operations without any performance drawbacks.



## CHAPTER 4

# SERVER REACHABILITY BEHIND NAT

The use of NAT introduces a critical problem for servers. Because NAT routers appear as single hosts to the Internet, NAT prevents certain end-to-end connectivity scenarios between hosts. It allows internal clients to initiate connections to external servers, but it does not allow external clients to reach internal servers. In this chapter, this issue is discussed, and solutions for HTTP and SMTP servers are proposed together with a performance evaluation of these solutions.

### 4.1 NAT Impact on Server Reachability

A server on the Internet is addressable using a tuple of its IP address and port. Any client can reach such server using this tuple. Normally, servers have public IP addresses, and thus, they are directly reachable. However, introducing NAT changes the IP address of the server to a private IP address, and the server is only

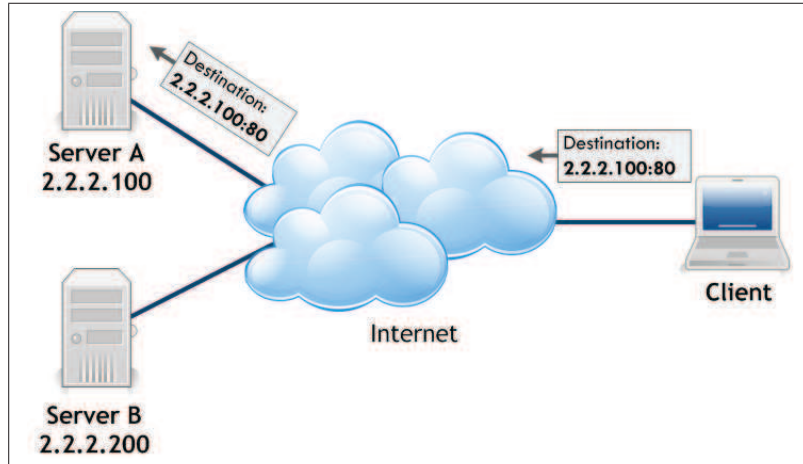


Figure 4.1: HTTP Request sent to a server with a public IP address

seen using the NAT router's public IP address. Moreover, running multiple servers for the same service, like HTTP servers, behind a single NAT router means that these servers are sharing the public IP address. Therefore, they are all addressable using the same tuple: NAT public IP address and the service port.

Figure 4.1 shows two HTTP servers with public IP addresses, connected to the Internet through a router. A client sends an HTTP request to server *A* using its public IP address (2.2.2.100) and HTTP port (80). The router uses the network-layer information, i.e., the IP address, to forward the request to the correct server. Since the server's IP address is public and thus unique, there is no ambiguity as to which server should receive the request, and the router would never send the request to server *B*.

On the other hand, if the router in the previous example is replaced with a NAT router, as shown in figure 4.2, the servers (*A* and *B*) and the router appear as a single host to the Internet, with the NAT's public IP address (3.3.3.100). The

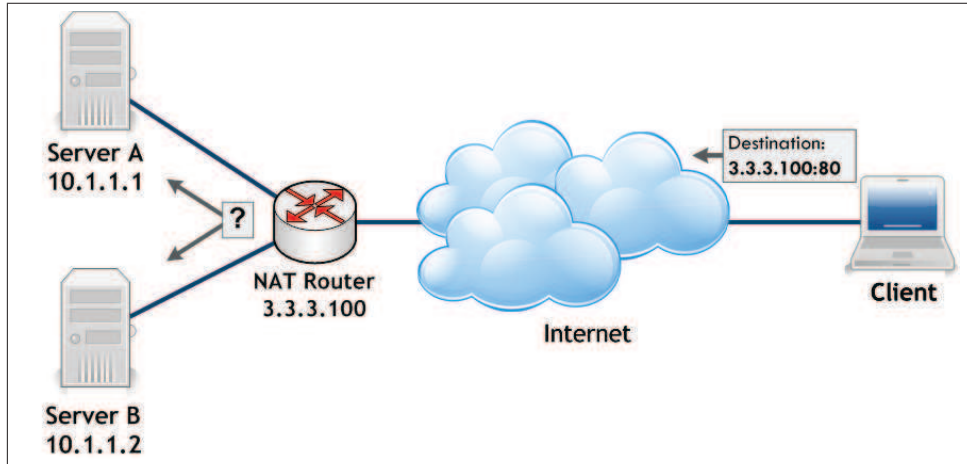


Figure 4.2: HTTP Request sent to a server behind NAT. The NAT router is unable to identify what is the correct destination of the packet

IP addresses of the servers are now private, and any request to the servers would have the NAT's public IP as its destination IP address. An incoming request to server A, for example, will have the destination tuple (3.3.3.100, 80). The request reaches the router, and the router is confused about which server should receive this request, because the NAT has no matching entry in the NAT table. The request will be dropped by the router, unless the router is manually configured to forward the traffic to a specific server.

## 4.2 Related Work

Running multiple servers with a single public IP address has been used in many web-server scalability designs. Web clusters and distributed web servers, are the most common examples of such design. Some approaches are used to run a single website on multiple servers with a single IP address to achieve load

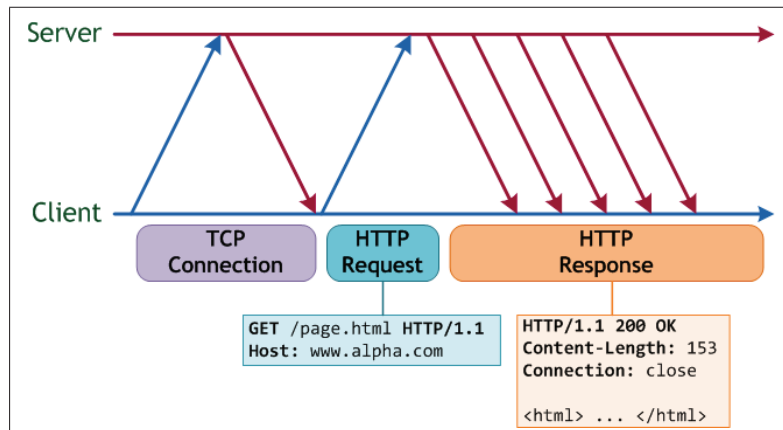


Figure 4.3: The three steps of HTTP communication are: TCP connection, HTTP request, HTTP response

balancing, while other approaches are used to run multiple websites on a single server with one IP address to achieve higher utilization of hardware.

#### 4.2.1 Multiple Websites on a Single Web Server

A very common technique to run multiple websites over a single server with a single IP address is *Virtual Hosting*. This technique uses layer-7 information, specifically HTTP request headers, to specify which site is the correct destination for that request. To explain the technique further, we first explain how HTTP works.

Any HTTP communication consists of three steps: the client initiates a TCP connection, it then sends the HTTP request, and the server sends back the HTTP response. Figure 4.3 shows an example of HTTP communication between a client and an HTTP server.

An HTTP request consists of three parts:

1. A request line which defines the method of the request, the address of the requested resource, and the protocol version.
2. HTTP headers, meta-data that define characteristics of the data that is requested.
3. A message body, used optionally to send data to the server.

The *Host* header is an important HTTP header field. It was first proposed by John Franks [56] to allow a server with a single IP address to host multiple websites. It was later standardized in HTTP version 1.1 as a mandatory header field [57].

The *Host* header defines the host name of the destination website. For example, a web server hosts two sites, `www.alpha.com` and `www.beta.com`, as shown in figure 4.4. A client initiates a TCP connection, then sends the following HTTP request:

```
GET /page.html HTTP/1.1
Host: www.alpha.com
```

The web server receives the request, and by looking at the *Host* header, it can decide which website is the requested one. This basically means that application-layer information is used to identify the destination service.

The *Virtual Hosting* technique is one of the most popular hosting options available today, due to its cost-efficiency as only one server, with one IP address, is needed to host many web sites. It is implemented in most HTTP server applications, such as Apache [58] and Microsoft's IIS [59].

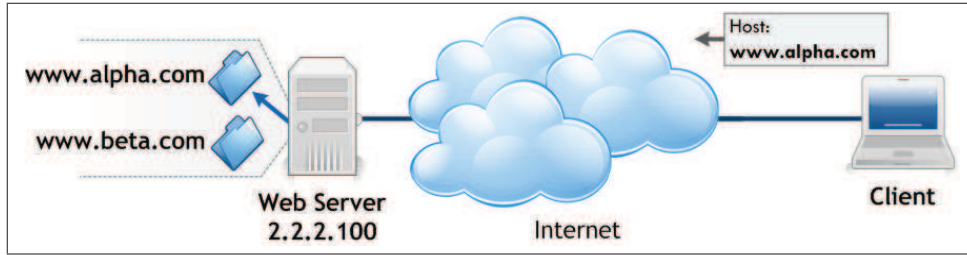


Figure 4.4: In virtual hosting, The web server uses the HTTP Host header to determine which web site is the correct destination for the request

## 4.2.2 Multiple Web Servers With a Single IP Address

Cardellini et al. [60] have made a complete survey over the existing distributed web-server systems. They classify these systems into two categories: *web clusters*, where the server nodes are sharing the same IP address and the web cluster is seen by the client as one host, and *distributed web systems*, where the IP addresses of the server nodes are visible to the client. The first category of systems, i.e., *web clusters*, is the one related to our problem, as it allows multiple servers to use a single IP address.

The techniques researched in Cardellini's work are based on having a number of nodes that act as web servers. These nodes are connected to the Internet through a router that is responsible for the request routing and dispatching. In web clusters, the server nodes share the same public IP address, called *Virtual IP* (VIP). This address is the public IP address of the router connecting the server nodes to the Internet.

There are two types of routing mechanisms for web clusters: layer-4 routing and layer-7 routing. In layer-4 routing, the router is content-blind, i.e., it is not

aware of the application-layer information such as the requested page. Therefore, all server nodes have the complete content of the website. On the other hand, layer-7 routing is content-aware. Hence, it is possible to distribute the content over different server nodes, where each node can serve a specific type of content. This type of routing is more sophisticated, however, it provides more room for content-distribution and load-balancing.

In layer-7 routing, requests are first accepted by the router, which can perform layer-7 operations. This router is also called *web switch*. The web switch accepts the TCP connection, receives the HTTP request, then decides which server node should handle this request based on some dispatching policy. Cardellini discussed two ways the web switch can handle the request to the selected node:

- **TCP gateway:** The web switch acts as an application-layer proxy to mediate the communication between the client and the server. This proxy accepts client connections and maintains TCP persistent connections with all the server nodes. When a request arrives on a client connection, the proxy forwards the client request to the target server through the corresponding TCP persistent connection. When the response arrives on the persistent connection back from the server, the web switch forwards it to the client through the other connection.
- **TCP splicing:** This technique aims to improve the TCP gateway technique that is computationally expensive. Here, packet forwarding occurs at the network-layer between the network interface driver and the TCP/IP stack,

and is carried out directly by the operating system on the router and the server node. Once the TCP connection between the client and the web switch has been established and the persistent TCP connection between the switch and the target server has been chosen, the two connections are spliced together. IP packets are forwarded from one endpoint to the other without having to cross the TCP layer up to the application layer on the web switch. Once the client-to-server binding has been determined, the web switch handles the subsequent packets by changing the IP and TCP packet headers (IP addresses and checksum recalculations), so that both the client and the target server can recognize these packets as destined to them.

Although the *TCP gateway* technique introduces larger computational overhead, it has the advantage of not requiring any changes to be made in the server nodes. In *TCP splicing*, the server nodes must be modified to be able to handle spliced connections. However, *TCP gateway* can run transparently, and server nodes do not need to be modified.

Cardellini's research on web clusters is based on techniques to run a single website over a number of servers with a single IP address. This is different than our problem, where different websites are ran on different servers, but they all share the same IP address. Therefore, the web clustering techniques do not fit as solution to the servers-behind-NAT problem. However, one of the discussed layer-7 routing techniques, namely the *TCP gateway* technique, will be utilized to develop a solution to run servers behind NAT.



Because the NAT router is unable to determine which server is the correct destination for a received request, the solution is basically to define a method to identify the destination server for a given request.

## **4.3 Proposed Solution for HTTP and SMTP Servers**

The deployment of the NAT-based solution for Internet denial results in preventing external clients from accessing the services within the private network. Therefore, we develop two solutions for HTTP and SMTP servers, since they are the most commonly used services.

### **4.3.1 HTTP Servers Behind NAT**

#### **Server Reachability Issues**

When the NAT router receives a request for a web server behind it, it is unable to identify the correct destination for that request. There are many web servers behind the NAT router; all of them are sharing the same address, i.e., the NAT public IP address, and they also share the same TCP port, the standard HTTP port, 80. There is no network- or transport-layer information that tells the NAT router which server is the destination.

The problem, therefore, is that neither network-layer information nor transport-layer information is usable to identify the correct destination of the

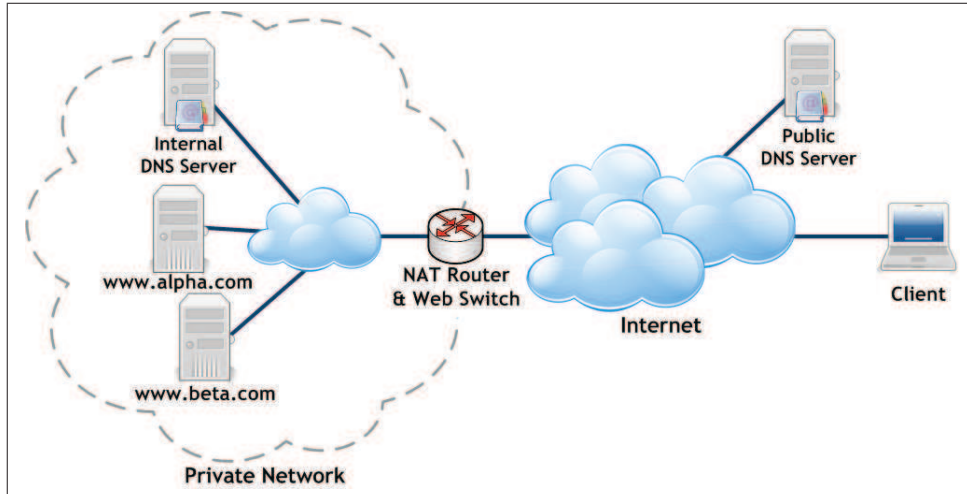


Figure 4.5: Initial setup for the solution of HTTP servers behind NAT uses a web switch for intermediate request forwarding

HTTP request. However, we have seen that application-layer information, namely the HTTP *Host* header, can be used to map requests to their destination websites. Hence, the solution we propose is based on using a similar concept, but on a server-level mapping, rather than a website-level mapping.

### Solution Setup

The initial setup for the proposed solution, as shown in figure 4.5, includes the NAT router, connected to the Internet with a public IP address, a number of web servers in the NAT's private network, and a client that is connected to the Internet and is attempting to access one of the web servers behind the NAT router. There is also a DNS system consisting of the public DNS servers that the client uses to determine the IP address of the servers, which is basically the NAT's public IP address; and the private DNS servers that are used by the clients within the private network, as discussed earlier in section 3.3.3.

## Use of Web Switch

A *web switch* is used to accept HTTP request. It can either replace the NAT router, acting as both NAT router and web switch, or the NAT router can just forward all traffic destined to port 80 to the web switch behind it.

Public DNS servers must be updated with the new IP addresses. In this case, the NAT's public IP address is used for all servers. Hence, entries in the public DNS servers are updated with the NAT's public IP, which we assume is not blocked by the malicious ISPs. The clients will use the public DNS servers to resolve the domain names of the HTTP servers to the NAT's public IP address. The management of IP addresses will be discussed later in this chapter.

## Identifying the Correct Destination

After resolving the server's name and getting the IP address, the client initiates a TCP connection to the HTTP port, 80, of that IP address. The web switch accepts the connection and receives the HTTP request. Then, the web switch reads the *Host* header from the received request, and uses the internal DNS servers to resolve the host name into an IP address. This IP address, corresponding to the private address of the correct destination server of the request, is directly accessible by the web switch since the switch is part of the private network.

Now, the web switch has identified the correct web server for that HTTP request. The next step is to forward the HTTP request to that server. The proposed technique to perform that is to combine *TCP gateway* [60] with NAT

tables to provide a transport-layer forwarding of traffic.

### **Forwarding Requests to the Server**

After the web switch receives the HTTP request and identifies the intended web server, an entry is added to a special NAT table that maps the client's address tuple (client's IP address and source port) to the server's address tuple (server's private IP address and destination port, 80 in this case). This table entry is used to forward subsequent traffic between the client and the web server until the end of the HTTP session. Similar to the usual NAT tables, the entries are deleted after some timeout period, or at the disconnection of TCP connection using a packet with the FIN flag.

### **Example of Accessing a Web Server Behind NAT**

To illustrate how the process takes place, figure 4.6 shows an example of a request sent to the server *www.alpha.com* behind the NAT router. The client sends the request to the NAT's public IP address, 3.3.4.4. The web switch will first accept the TCP connection, start receiving the HTTP request until it receives the *Host* header. The request can be sent in more than one packet, therefore, the router does not need to wait until the complete request is sent; it can start finding the correct server once the *Host* header is received.

Finding the correct server can be done by performing a DNS lookup in the internal DNS server within the NAT. This server should allow resolving all local domain names into the corresponding private IP addresses. Once the server's IP

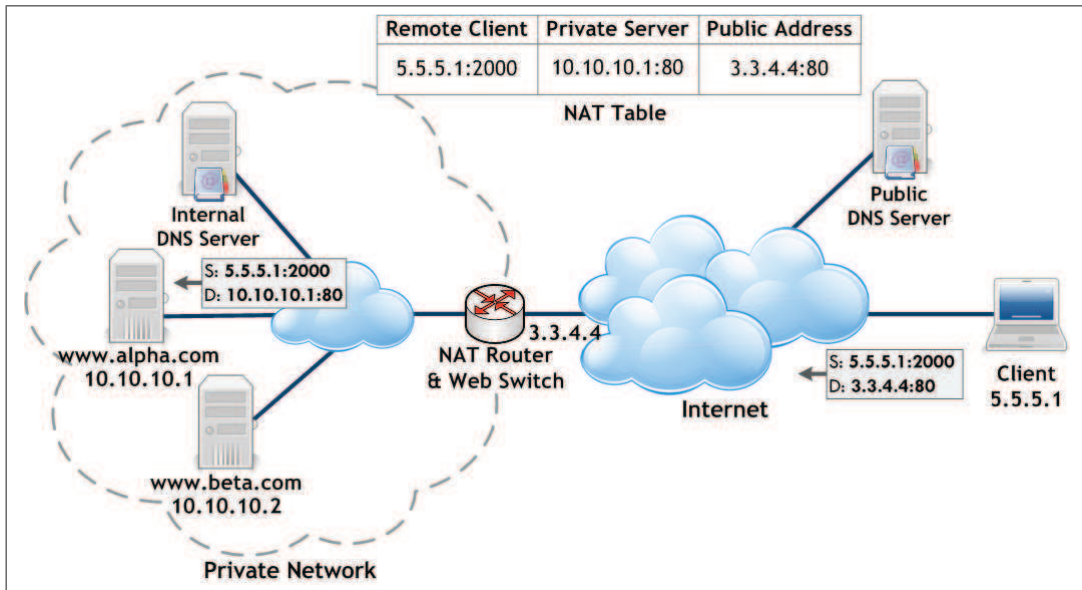


Figure 4.6: Example of using the HTTP *Host* header to access a server behind NAT

is found, 10.10.10.1 in this example, the NAT router adds an entry in its table. The table entry contains the client's IP address and port (5.5.5.1:2000), and the server's private IP address and port (10.10.10.1:80). This entry is used to map subsequent traffic, both request and responses, to the corresponding client and server.

After finding the server and adding an entry to the mapping NAT table, the web switch can now send the request to the server. The packets of the request need to be address-translated, by replacing the destination IP address, which originally was the NAT's public IP address (3.3.4.4), with the private IP address of the server (10.10.10.1). The responses sent from the server to the client are also translated, replacing the server's source IP address (10.10.10.1) with the NAT's public IP address (3.3.4.4). This is very similar to the typical way NAT works. It

can be thought of as *reversed NAT*, where the traffic is initiated outside the NAT and destined to hosts within the NAT.

The advantage of using this technique is that no modifications are made to the servers within the NAT. Hence, using this technique together with the NAT solution for *Internet denial* would be transparent from the servers. Adding the NAT layer between the private network and the public Internet does not affect most of the web servers within the private network.

Some web servers, namely HTTPS servers, will still be unreachable even when the proposed technique is used. The reason is that HTTPS runs the HTTP protocol over an encryption presentation layer, which causes all application-layer information to be encrypted before transmission. Therefore, the web switch would not be able to read layer-7 information that is needed to determine the intended server, and the server is, hence, unreachable.

## **Design Scalability**

Because the solution can be used for very large networks with high incoming traffic, its design should be scalable to handle such load. The web switch is the bottleneck of this solution. Hence, better scalable design is needed.

One approach is to use load-balancing. Incoming requests are distributed equally over a number of web switches that are interconnected with the gateway NAT router. Each web switch uses the described technique to forward incoming HTTP requests to their intended destination servers. This design is shown in figure 4.7.

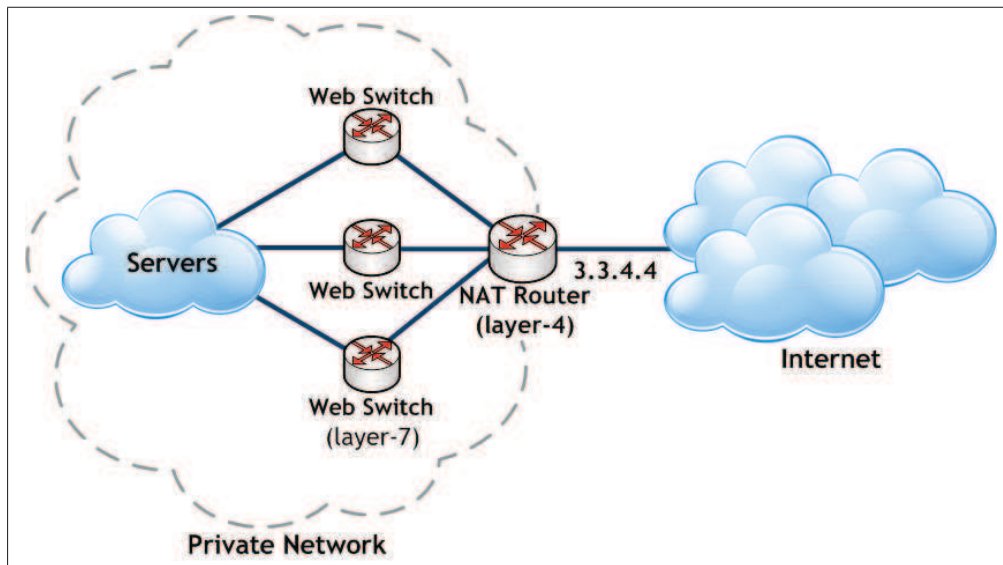


Figure 4.7: Incoming request are load-balanced over a number of web switches to increase scalability

When a request is received by the NAT router, it first checks its NAT table to see if this request has already been mapped to one of the web switches. If no mapping is found, the NAT router selects one of the web switches to handle this request, adds an entry to its NAT table to map the connection with that switch, and then forwards the packets to the selected web switch. The processing done at this level is only layer-4 processing, no application-layer data are processed yet.

The selected web switch accepts the client's request, and performs the described technique of finding the correct server using layer-7 information. It then forwards the request to the intended server after adding an entry on its application-layer NAT table.

The objective of the NAT table at the NAT router is to map web switches to incoming packets. Subsequent packets from the same client destined to the same server should all be processed by the same web switch. Hence, the NAT table is

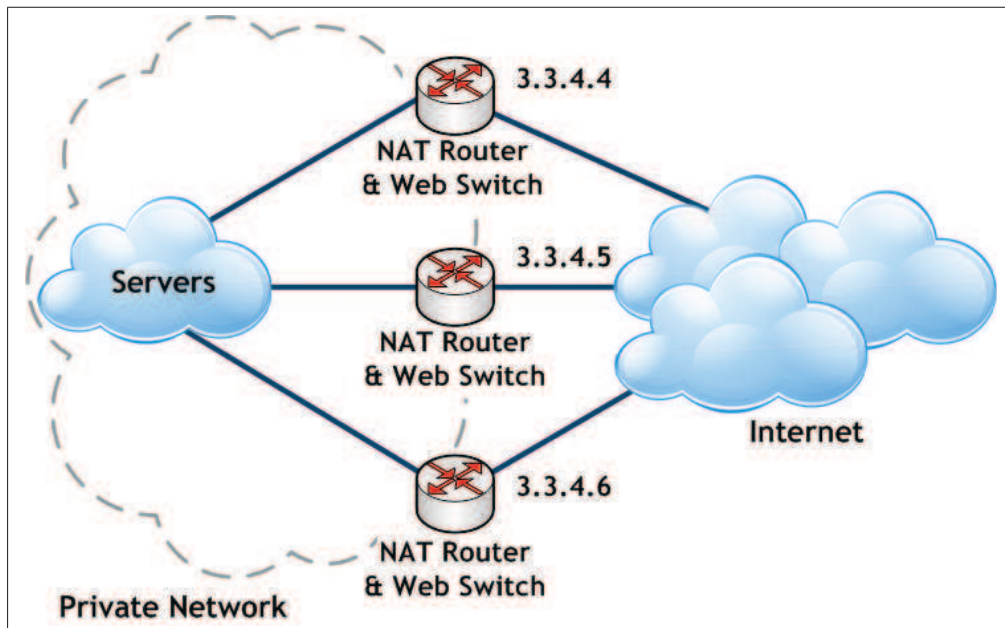


Figure 4.8: DNS is used to distribute the load over web switches

used to keep track of this mapping.

On the web switch, the table is used to map packets of the same connection together. Once the server is located using layer-7 information, all subsequent packets are forwarded directly to the web server, and all responses are forwarded directly to the client.

The other approach of load-balancing the incoming traffic is to utilize DNS. Entries in the DNS can have more than one IP address. Hence, clients will use different IP addresses to connect to the same server. By placing a number of web switches at the gateway level, each with a different IP, incoming traffic will be balanced over the different web switches. Figure 4.8 shows the topology used to implement this approach.

It is possible to combine both approaches to maximize scalability. A number



of NAT routers can be used at the gateway level, each with a different IP address. These IP addresses are all used in the DNS for load-balancing. Each NAT router is connected to a number of web switches that will process layer-7 information and forward the requests to the intended destination servers. This way, load-balancing is performed at both gateway level and web switch level.

### 4.3.2 SMTP Servers Behind NAT

#### SMTP: How It Works?

Simple Mail Transfer Protocol (SMTP) is the standard protocol that is used to deliver email messages on the Internet [61]. The SMTP email system is formed of email servers, client agents, and the email protocol, SMTP. Email servers allow users to connect to them through client agents in order to send and receive emails. The servers keep incoming email messages in the users' mailboxes so that users can retrieve them later using the client agent application. Outgoing messages are sent from the client agent to the local email server, and then from the local server to the destination email server where that server stores these messages in the recipient's mailbox. SMTP is used for the exchange of messages between mail servers, and for sending email from the client agent to the email server. Figure 4.9 shows how SMTP is used to deliver email messages from the user agent to the destination server. The email message is first sent from the user agent to the local email server using SMTP. The server, then, uses SMTP to deliver the email to the destination server, *beta.com*. Once delivered, the client can retrieve the

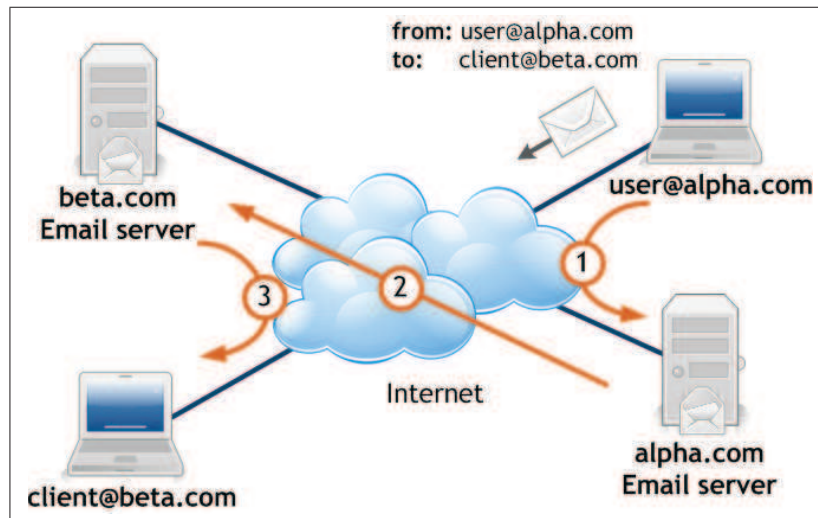


Figure 4.9: SMTP protocol is used to deliver messages to email server

email message using email access protocols, such as *Post Office Protocol* (POP) or *Internet Message Access Protocol* (IMAP).

An email address is written in the form *account@domain*, which consists of two parts: the account name, which specifies the destination user of the message; and the domain name, which specifies the server that maintains that user's account.

When a mail server receives an email message, it looks at the recipient's email address to determine the destination domain name. It then uses DNS to determine the IP address of the *mail exchange* (MX), the email server that handles SMTP services for that domain name. DNS has a special entry type called MX, which is designated for mail exchanges. After the server finds the destination IP address, it sends the email message to it using the SMTP protocol.

## **Solution Setup**

The solution for reaching SMTP servers behind NAT is less complex than the one for HTTP, simply because SMTP always uses the domain name to determine the destination email server.

The initial setup for this solution is shown in figure 4.10. The solution uses an SMTP relay, which is an intermediate SMTP server that acts as a relay to deliver email messages. The NAT router should forward all SMTP traffic, i.e., TCP connections destined to the standard SMTP port 25, to the SMTP relay.

## **Use of SMTP Relay**

When an external SMTP server tries to connect to an SMTP server within the private network, it first uses DNS to lookup the MX entry for the destination server. Therefore, MX entries in the public DNS servers must be updated with the SMTP relay's IP address, which is basically the NAT's public IP address.

After the source SMTP server resolves the domain name and finds the IP address of the SMTP relay, it initiates a TCP connection to send the messages. The NAT router receives the connection destined to port 25, and forwards it directly to the SMTP relay. The relay accepts the connection, and receives the messages. Then, it queues them to be sent to their final destinations. The SMTP relay uses the internal DNS servers in order to resolve the domain names and find the private IP addresses of the destination mail servers. It then establishes connections to these servers and deliver the email messages.

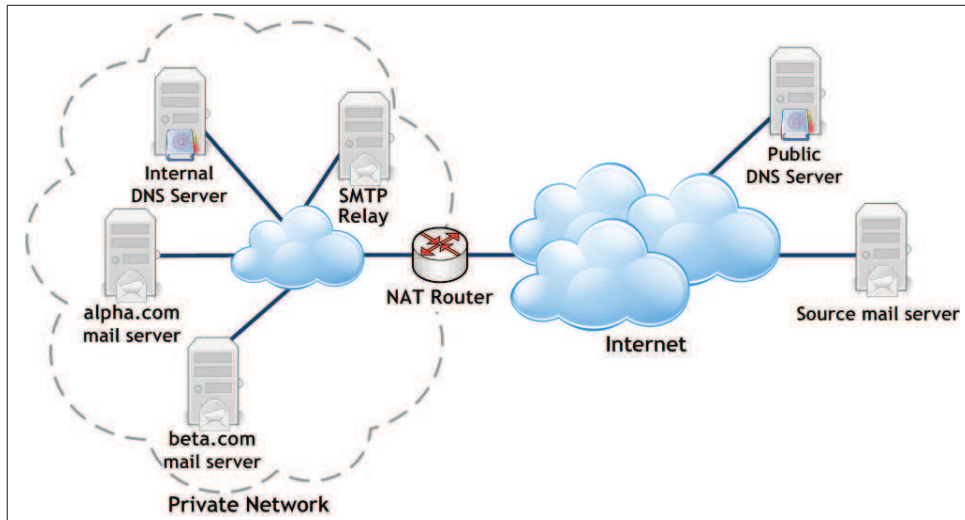


Figure 4.10: Initial setup for accessing SMTP servers behind NAT uses an SMTP relay server as an intermediate message delivery point

### Example of Delivering an Email Message

For example, an external mail server attempts to deliver an email message to *user@alpha.com*. The server will lookup the MX entry for *alpha.com*, using a public DNS server, then it connects to the found IP address. That IP address belongs to the NAT router, which is configured to forward all incoming packets destined to the SMTP port 25, to the SMTP relay.

The SMTP relay accepts the connection, then receives the email message though SMTP. After that, the relay uses the internal DNS server to lookup the MX entry for *alpha.com*. The private IP address of that server is found, and the relay establishes a connection to the server *alpha.com* and delivers the email message.

## 4.4 Management of IP addresses

The proposed NAT solution for the Internet denial is based on hiding the identity of traffic in order to deceive the malicious ISPs to route the traffic. The IP addresses that are used for NAT belong to a different network that the malicious ISPs are not blocking. We assumed earlier that the ISPs will block the IP address blocks only once. That is, malicious ISPs will not try to detect or block new IP addresses that the victim network may use after the initial blocking. However, it is very important to keep the NAT IP addresses unexposed to the malicious ISPs. It should not be evident to the ISPs that the new IP addresses are being used by the blocked network to bypass the blocking. Therefore, there is a trade-off between two approaches to manage the new, unblocked public IP addresses.

The first approach is to use the available IP addresses for both local clients and servers, as shown in figure 4.11. The IP addresses that are used in the NAT router to map outgoing traffic are also used as addresses for the HTTP and SMTP servers. Domain name and MX entries in the public DNS servers are updated with these IPs. This approach provides better utilization of the available IP address space, and therefore, requires less number of IP addresses to be used for the solution. However, because public DNS servers are updated with these IP addresses, malicious ISPs can detect that services within the blocked network are using new IP addresses. Consequently, the malicious ISPs will block the new IP addresses, and the whole victim network, including both clients and servers, is blocked again.

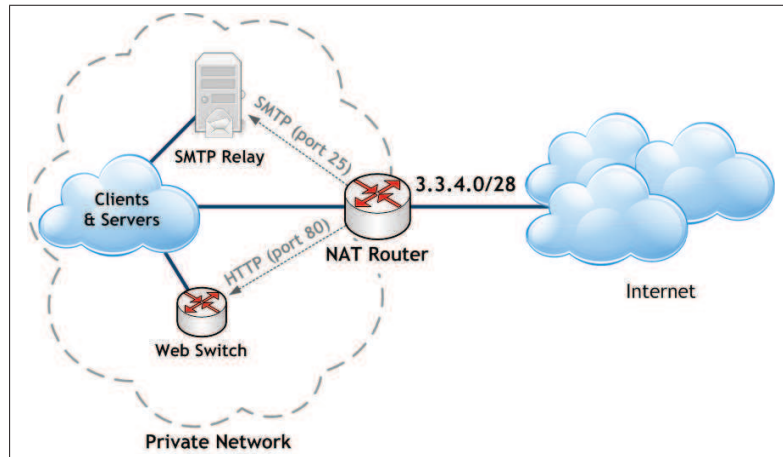


Figure 4.11: Using the same IP addresses clients and servers

The other approach is to separate the IP address blocks for clients and servers. As figure 4.12 shows, the NAT routers can use a pool of IP addresses to map outgoing traffic, and the servers are mapped to a different pool of IP addresses. Entries in the DNS servers are updated only with the server IP addresses. Hence, the IP address block that is used by the clients is not exposed to the malicious ISPs. Even if the malicious ISPs detected that the servers are using new IP addresses, blocking these IP addresses will only make the servers unreachable, but the clients would still be able to use the Internet. Although this approach has higher costs as it requires more IP addresses, it provides better security for client connectivity.

The assumption that the malicious ISPs will not perform any subsequent blocking of IP addresses makes both approaches similar in terms of security, but the first approach is better in IP address utilization. However, depending on the motivations of malicious ISPs, it is probable that they will block any new IP addresses

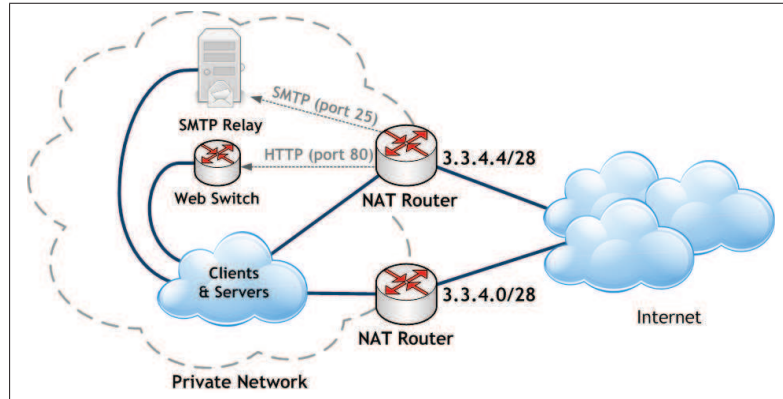


Figure 4.12: Using separate IP addresses for clients and servers

that belong to the victim network. Hence, the second approach provides better robustness against being blocked again, as only a small portion of the IP addresses might be blocked.

## 4.5 Performance Evaluation

The proposed solutions for HTTP and SMTP servers add extra overhead to the network, and therefore, have some impact on the performance. Simulations are used to evaluate the performance of the proposed solution for the HTTP servers behind NAT. The SMTP solution, on the other hand, is not simulated. The reason is that the delivery of email messages over SMTP protocol is not time-critical. The additional delay caused by the use of SMTP relays would only effect the end-to-end delivery time of the message. This delay, nevertheless, is not critical for an application like SMTP.

The objective of simulating the HTTP solution is to measure the impact of implementing the web server solution on the network performance. Two metrics

are used for measurements: end-to-end delay between the clients and servers, and the throughput of the sent and received traffic.

#### **4.5.1 Modeling of Web Switch Delay**

The processing done in the web switch is similar to the one NAT does, but in a reversed direction. The web switch maps incoming connections to their corresponding servers. The use of NAT table, and the translation of addresses and ports, are very similar to NAT. The proposed solution requires layer-7 processing of only the first packet, and subsequent packets are processed at layer-4. Hence, it can be assumed that the performance evaluation of NAT is a good approximation of the performance of a web switch, except for the layer-7 processing.

In order to measure the effect of layer-7 processing on the performance, the web switch delay is implemented in OPNET such that it adds an extra processing delay for the first packet of a request. Subsequent packets only suffer from the layer-4 NAT delay, which is smaller than the web switch delay. The implementation is done such that when a NAT table lookup is added, the packet processing delay is increased by the web switch delay. This is because NAT table entries are only added for the first packet of every session, which is the same packet that will have layer-7 processing. The processing time of all subsequent packets and responses only suffers from a small extra NAT delay.



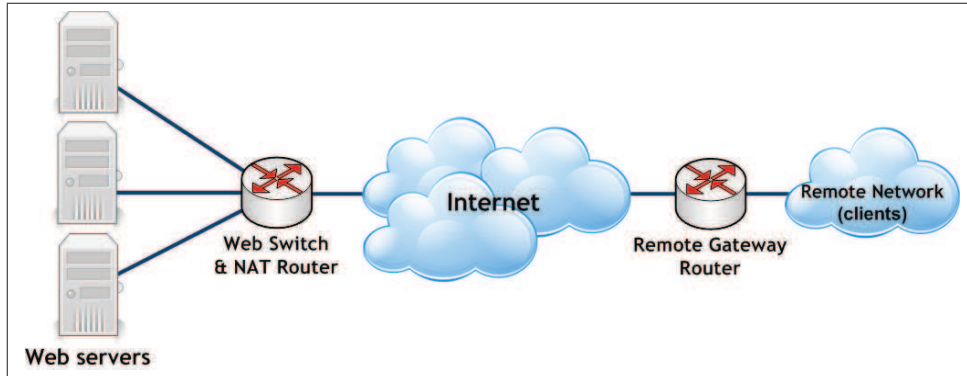


Figure 4.13: Simulated scenario for the HTTP server reachability solution

## 4.5.2 Simulated Scenario

The simulated scenario is similar to the one used in NAT evaluation, as shown in figure 4.13. The scenario consists of two networks, local and remote. Each network is connected to the Internet cloud by a gateway router. The local network consists of 3 web servers, and the local gateway is a web switch that has the implementation of NAT delay and web switch delay. The remote network consists of a 100BaseT Fast Ethernet LAN, with 10 hosts that act as web clients. The intermediate links, connecting gateway routers to the Internet, are DS-1 links with 1.544 Mbps data rate.

The measurements are selected to compare the performance of using a normal router, where servers have public IP addresses, with the use of a web switch, where packets suffer added NAT and web switch delays.

### 4.5.3 Simulation Setup and Parameters

Because all packets that pass through the web switch are translated, NAT delay is added to the processing time. It was shown earlier that the realistic range for simulated NAT delay is between  $10\mu s$  and  $50\mu s$ . We select  $50\mu s$  as the simulated NAT delay, to measure the impact of the worst-case scenario.

It was shown earlier that the significant degradation of performance caused by NAT is very similar for low and medium traffic. Therefore, we will only simulate two traffic scenarios: low, where 25% of the bandwidth is utilized (about 380 kbps), and high, where 75% of the bandwidth is used (about 1,200 kbps).

The web switch delay is caused by the processing of layer-7 information. Therefore, this delay is expected to be higher than the one caused by NAT for layer-4 processing. In the simulations, we vary the amount of web switch delay from  $100\mu s$  to  $400\mu s$ . This delay is only added to the first incoming packet of the session. Figure 4.14 shows the simulated end-to-end delay of a single client sending requests to the servers. It is noticed that at the beginning of each session, a small increase of the end-to-end delay occurs, which is the added web switch delay.

### 4.5.4 Simulation of End-to-end Delay

The impact of web switch on end-to-end delay is simulated. The delay is measured as the time a packet takes to traverse from the client to the server, including the transmission, queuing, NAT, and web switch delays.

Figure 4.15 shows the simulated end-to-end delay for the low traffic scenario. It

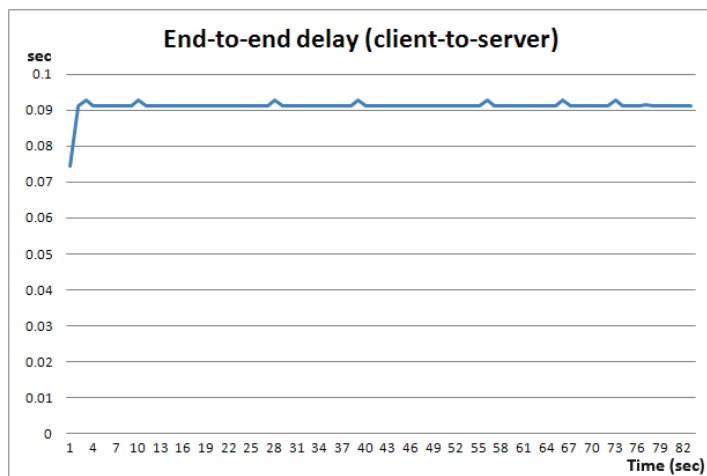


Figure 4.14: End-to-end delay for low web traffic

is noticed that a small increase is caused by the web switch. The effect increases as the web switch delay is increased. About  $100\mu\text{s}$  additional delay is measured when the web switch delay is  $100\mu\text{s}$ , but it increases by  $300\mu\text{s}$  for a web switch delay of  $400\mu\text{s}$ . There are two factors causing this increase. First, all packets require extra processing time because of the added NAT delay. Second, some packets, namely the first packet of every HTTP session, suffers an extra web switch processing delay. These two factors cause the increase of the end-to-end delay.

For the high traffic scenario, figure 4.16 shows the measured end-to-end delay from the client to the server. Similar to the previous scenario, the end-to-end delay increases when a web switch is used because of the added web switch and NAT delays which cause all packets to require extra processing time.

Figure 4.17 shows the amount of end-to-end delay increase caused by the introduction of a web switch. This increase is computed as  $(\text{Delay}_{\text{WebSwitch}} - \text{Delay}_{\text{Router}})$ . It is noticed that the increase in the low traffic scenario is in the

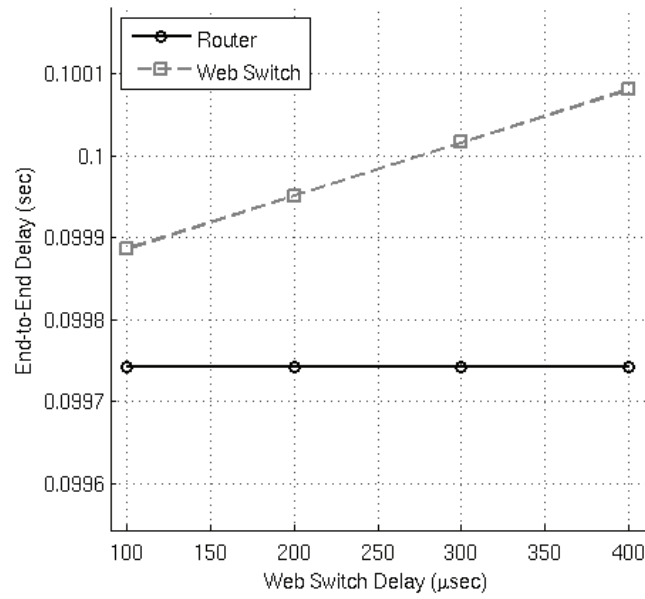


Figure 4.15: End-to-end delay for low web traffic

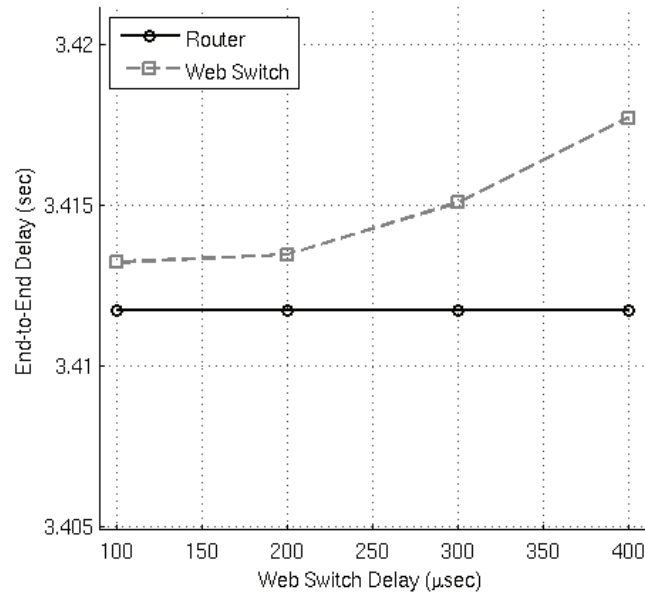


Figure 4.16: End-to-end delay for high web traffic

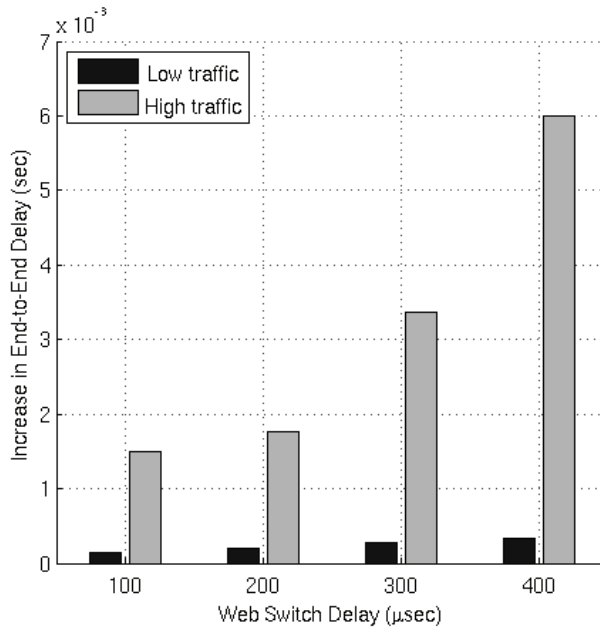


Figure 4.17: Increase in end-to-end delay for web traffic

order of hundreds of microseconds, whereas the increase in the high traffic scenario is in the order of milliseconds.

The relative increase, computed as  $\frac{(Delay_{WebSwitch} - Delay_{Router})}{Delay_{Router}}$  is shown in figure 4.18. Although the amount of increase in high traffic scenario is higher, the relative increase is smaller than the low traffic scenario. This has also been noticed earlier when NAT is simulated. The reason, as mentioned earlier, is that the processing delay for high traffic becomes less significant than the queuing delay. Hence, the relative increase, caused by NAT and web switch delay, is smaller.

We also notice that the maximum increase does not exceed 0.4% of the total end-to-end delay, which is similar to the results found in the simulation of NAT. This increase does not have significant effect on the performance. Hence, we can conclude that the proposed HTTP server solution has a small, insignificant impact

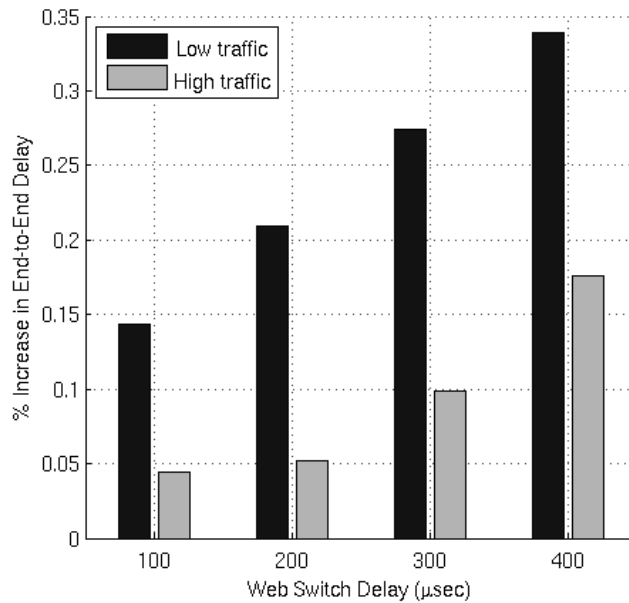


Figure 4.18: Relative Increase in end-to-end delay for web traffic

on the end-to-end delay.

#### 4.5.5 Simulation of Traffic Throughput

The throughput is measured in the simulation as the amount of traffic exchanged between the clients and servers. We measure the throughput as the traffic received on the client side.

In the earlier simulations of NAT delay, we noticed that NAT has *zero* impact on the throughput for low and medium traffic. The throughput is impacted only for the high traffic scenario.

The impact of the web server solution on the throughput is similar to the impact of NAT. The simulations show no impact on the traffic throughput in the low traffic scenario. However, the impact starts to appear in the high traffic scenario,

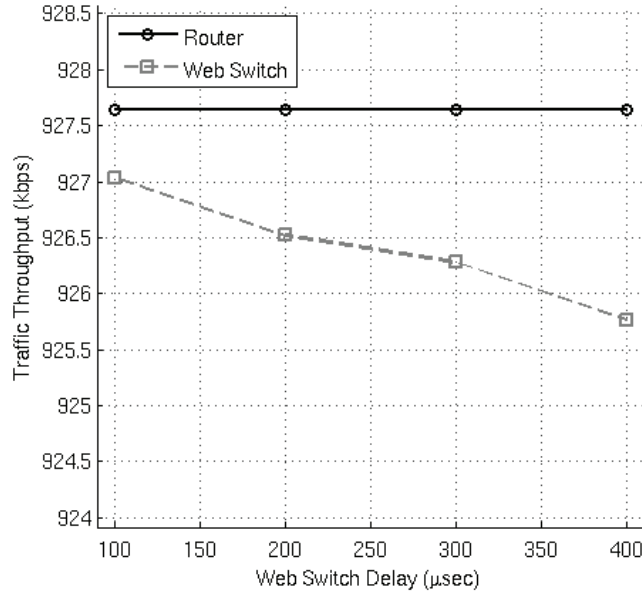


Figure 4.19: Throughput for high web traffic

as shown in figure 4.19. We notice that the throughput starts to decrease when the simulated web switch delay increases. This is because the added processing delay causes more packets to be queued in the web switch’s queue. Hence, The amount of transmitted traffic is lower.

The amount of throughput decrease caused by the introduction of a web switch is shown in figure 4.20. The decrease is computed as  $(Throughput_{Router} - Throughput_{WebSwitch})$ . We notice that the amount of decrease is relatively low; a maximum decrease of 250 bytes per second is experienced when the web switch delay is as high as  $400\mu s$ . Relatively, as shown in figure 4.21, the amount of decrease is limited to about 0.2% of the total throughput. This decrease is very low, and hence, can be considered negligible.

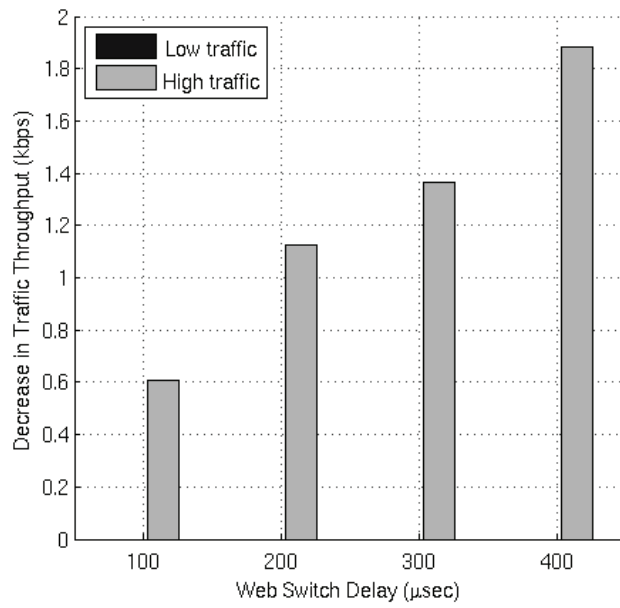


Figure 4.20: Decrease of throughput for web traffic

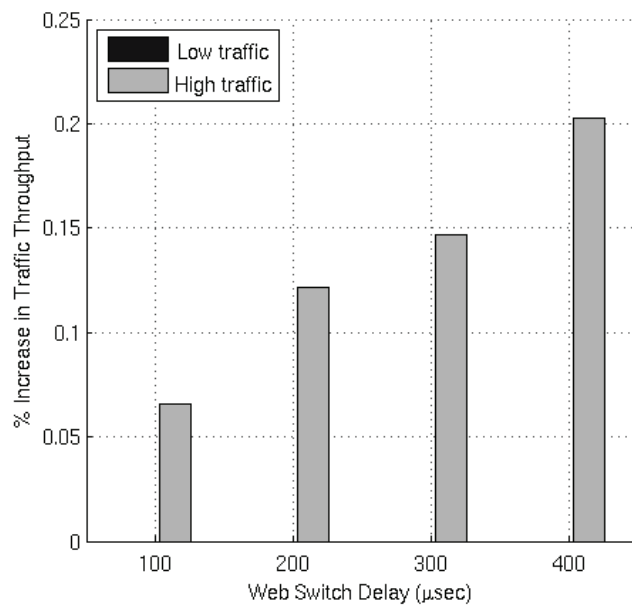


Figure 4.21: Relative decrease of throughput for web traffic



## 4.6 Conclusion

The simulation results show that the proposed solution for HTTP servers behind NAT does not cause any significant impact on the end-to-end delay or the throughput. The simulations were performed with the worst-case scenario parameters, i.e., high NAT delay and high web switch delay. Therefore, the realistic implementation of this solution on hardware would cause even less impact on the performance. Hence, we can conclude that the proposed solution has negligibly small impact on the performance of the network.

## CHAPTER 5

# PEER-TO-PEER CONNECTIVITY BEHIND NAT

In this chapter, we will look into the connectivity limitations introduced by NAT that affect peer-to-peer applications. Users in peer-to-peer networks must be able to initiate connections to remote users, and receive connections from other users. Because all users behind NAT appear to the Internet as a single entity, they cannot receive incoming connections. This impacts the peer-to-peer connectivity between users.

## 5.1 NAT Impact on Peer-to-Peer Applications

### 5.1.1 P2P vs Client-Server Applications

Network applications are classified into two types: *client-server* applications, and *peer-to-peer* (p2p) applications. In *client-server* applications, the client initiates the connection, sends the requests, and receives responses. A server is a centralized source that accepts connections and responds to requests. The service is only provided by the server, and all clients have to connect to that server to use this service. The World Wide Web (WWW) is an example of client-server applications, where clients must connect to servers in order to load pages and browse websites.

*Peer-to-peer* (p2p) applications, on the other hand, have decentralized services. Every *peer* in the p2p system is both a client and a server; it initiates connections to other peers, and accepts connections from others. Many file-sharing applications, such as Gnutella and BitTorrent, are p2p. Users on the network connect to each other to download and upload files. Figure 5.1 shows the difference between client-server and p2p applications models.

### 5.1.2 Lack of End-To-End Connectivity

One of the drawbacks of NAT is the limitation of end-to-end connectivity between hosts. This limitation prevents p2p applications from working properly. The reason is that a peer in a p2p network acts as both a client and a server. NAT only allows connections to be initiated from within the private network, and

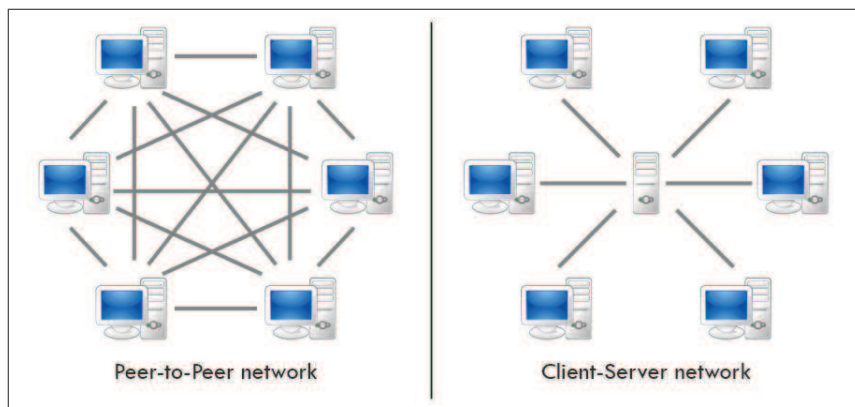


Figure 5.1: Peer-to-peer and client-server application models

destined to a host in the public Internet. Incoming connections are not received by the peer because there is no way of addressing the peer, as the private network behind NAT is seen by outsiders as a single host. Therefore, p2p applications will only act as clients, but not servers. This could work for some applications where the connections are initiated by the peers behind NAT. However, if the remote peer is also behind NAT, the problem is more complex.

### 5.1.3 Internet Denial Solution and P2P Applications

The introduction of NAT as a solution to the Internet denial problem results in connectivity issues for peer-to-peer applications. Moreover, users within the private network may have NAT routers at their local networks. Hence, introducing the NAT solution adds a new layer of NAT, resulting in multi-level NAT at the local user side. Peers on the public network may also have NAT on their own networks, adding a third level of NAT, as shown in figure 5.2. This scenario is more complex to resolve as it causes many p2p applications to stop working

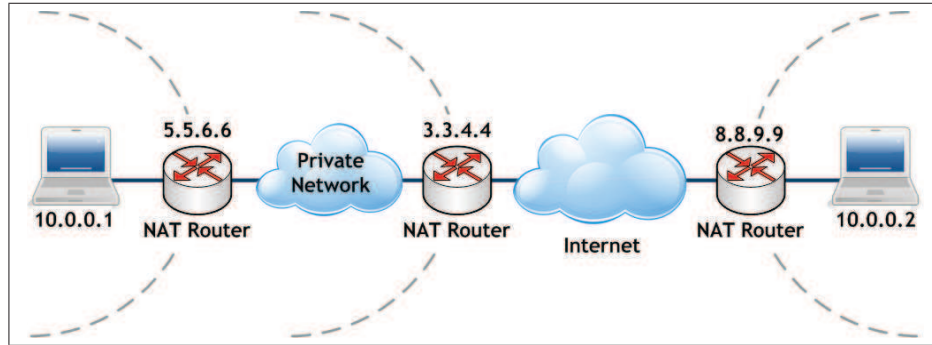


Figure 5.2: Multi-level NAT on both networks affects connectivity in p2p applications. Deployment of NAT solution adds additional level of complexity.

properly.

## 5.2 Related Work

Peer-to-peer applications have become very popular in the last few years. Studies show that p2p applications are responsible for more than 60% of the Internet traffic [62, 63]. The growth of p2p applications has been a motivation for many works that address p2p issues, such as connectivity, security, and performance.

### 5.2.1 NAT Behavior

The standard specification of NAT, like many other protocols' specifications, does not define all the implementation details. Some details are left to the vendors to implement in their way. Current NAT implementations differ not only from vendor to vendor, but also from model to model. Some of these differences can affect application connectivity. An application that works on one NAT environment may not work on another NAT environment. Many works have been done

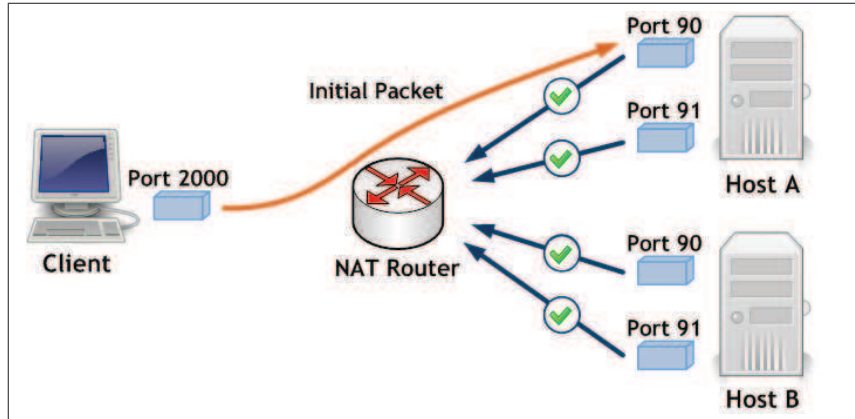


Figure 5.3: Full Cone NAT matches incoming packets using their destination port only

to discover and classify the behavior of NAT.

For example, *Simple Traversal of UDP Through NAT* (STUN) [64] was a protocol used to determine the behavior of NAT and classify it into one of four types:

- **Full Cone:** Requests from the same internal IP address and port are mapped to the same external IP address and port, even if the destination is different. Furthermore, any external host can send packets to the internal host, by sending the packets to the mapped external IP address and port. In other words, matching between the NAT table entries and the incoming packets are performed on the packet's destination IP address and port only. Figure 5.3 shows an example of how packets from different hosts and different ports are allowed to be forwarded to the client.
- **Restricted Cone:** Similar to a Full Cone NAT, requests from the same internal IP address and port are mapped to the same external IP address

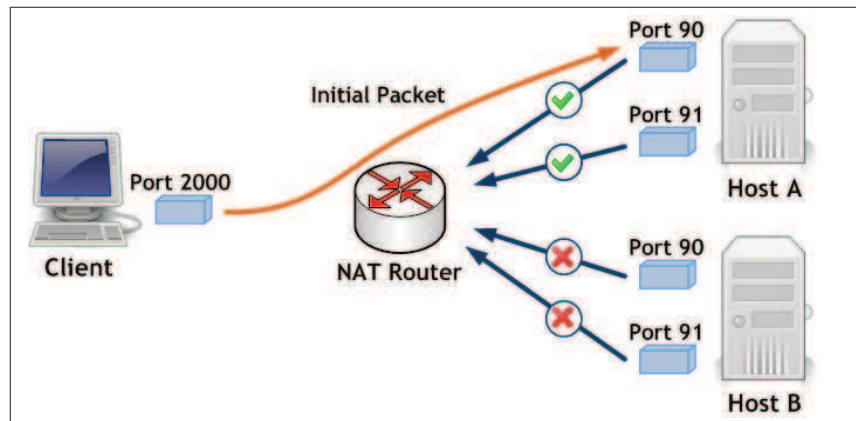


Figure 5.4: Restricted Cone NAT allows incoming packets only from the original destination host

and port. However, an external host *A* can send packets to the internal host only if the internal host had previously sent a packet to the IP address of host *A*. Hence, matching of incoming packets are performed on the packet's source IP address and destination IP address and port. Figure 5.4 shows an example of a Restricted Cone NAT.

- **Port-Restricted Cone:** It only differs from Restricted Cone NAT in the matching of incoming packets. It blocks all packets unless the client had previously sent out a packet to the IP and port pair that is sending to the NAT. Hence, the NAT table matching is performed on all the packet's address information, namely the destination IP address and port, and the source IP address and port. An example is shown in figure 5.5.
- **Symmetric:** This type of NAT differs from the first three in that outgoing packets are mapped uniquely. In Cone NATs, the same external IP address and port are used for all packets sent from the same internal IP address

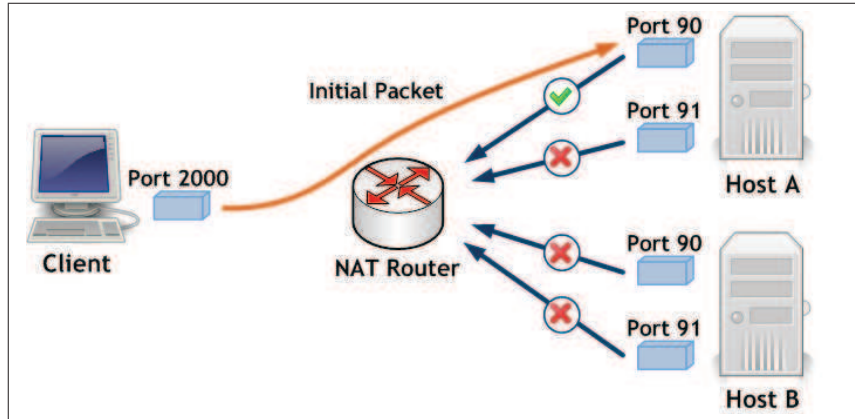


Figure 5.5: Port-Restricted Cone NAT allows incoming packets only from the original destination host and port number

and port. In contrast, packets in Symmetric NAT are mapped to different external IP address and port if they were destined to a different IP address or port. Figure 5.6 shows the difference in external address and port mapping between Cone NAT and Symmetric NAT.

This classification of NAT, though used in many protocols, was proven to be faulty, as many NAT environments do not fit cleanly into one of these types. Moreover, the behavior of some NAT routers is nondeterministic, as they may change their method of mapping and translation in some situations [65].

## 5.2.2 NAT Traversal Techniques

Much work has been done to resolve the NAT drawback of limiting connectivity. Different techniques and protocols were proposed as solutions to this problem. These solutions are called *NAT Traversal* techniques and protocols, as they are basically used to traverse packets over NAT.



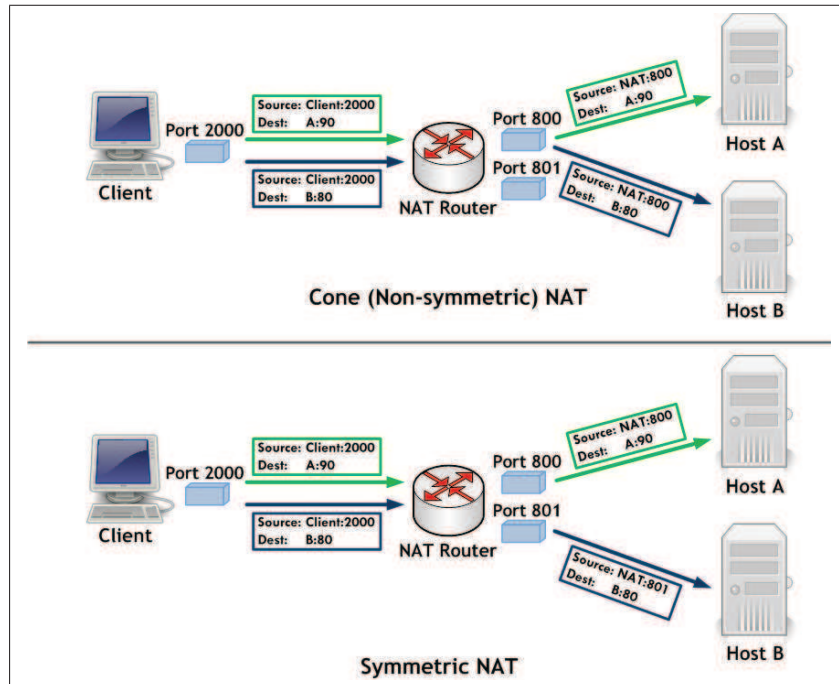


Figure 5.6: Cone and Symmetric NAT are different in the port mapping of outgoing packets

A *NAT traversal technique* is an approach for the process of establishing a connection between hosts that are located behind NAT. A *NAT traversal protocol*, on the other hand, is what defines how these techniques are used, and how the communication between hosts is performed. A NAT traversal protocol may use one or more NAT traversal technique in order to achieve connectivity.

NAT traversal techniques can be classified into two types: *control-based NAT traversal* and *behavior-based NAT traversal* [66]. The difference between them is whether the host explicitly cooperates with the NAT router to achieve connectivity. In control-based techniques, the NAT router provides cooperative functionality to help hosts in establishing and receiving external connections. On the other hand, the host in behavior-based techniques exploits the behavior of NAT in order

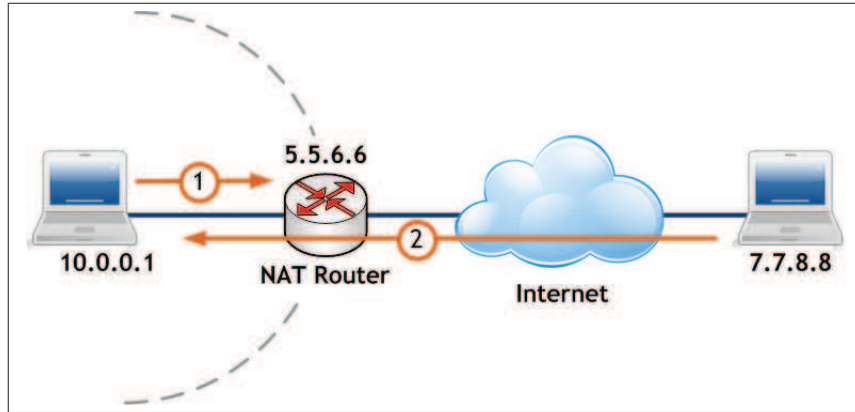


Figure 5.7: Control-based NAT traversal techniques allow clients to configure a port or socket to receive incoming traffic over it

to traverse traffic. The NAT router does not provide any additional functionality to help in traffic traversal.

### Control-Based NAT Traversal

In control-based techniques, the host communicates with the NAT router in order to allow connection establishment. These set of techniques are based on the idea that “NAT can completely and correctly be implemented only with the knowledge and help of the end hosts behind a NAT gateway” [67]. Hosts in the private network communicate with the NAT router to map specific ports or sockets, then they receive traffic over these mappings as figure 5.7 shows.

*Port forwarding* is a popular control-based technique. The NAT router allows the client to map a port so that all incoming connections destined to that port are forwarded to the client’s address. This allows the client to receive connections on the selected port. Almost all NAT routers allow manual port-forwarding, i.e., the router can be configured to forward traffic destined to a specific port to a static

host within the private network. Dynamic port-forwarding, on the other hand, requires the support of one or more port-forwarding protocols.

*Internet Gateway Device Protocol* (IGD) [68], also known as Universal Plug and Play (UPnP), is a popular port-forwarding protocol that is implemented in many routers. It allows hosts to learn the public IP address, and add port mappings to their addresses. *NAT Port Mapping Protocol* (NAT-PMP) [69] is a similar protocol that was introduced as a standard alternative to IGD.

*Realm-Specific IP* (RSIP) [70] is an experimental protocol intended to replace the behavior of NAT. It maintains end-to-end connectivity by allowing hosts within the private network to *borrow* public IP addresses and ports from the NAT router in order to communicate through them. Though it was proposed back in 2001, this protocol is still in the experimental stage, and is not widely deployed yet.

*Middlebox Communication* (MIDCOM) [71] defines the functionalities for *middleboxes*, intermediate devices that provide specific services, such as NAT and firewall. MIDCOM protocol modifies NAT functionality by moving the control of NAT resources, namely IP addresses and ports, to the protocol clients. Hence, clients can configure NAT dynamically to forward incoming traffic to them.

The other control-based technique for NAT traversal is traffic *gateways*. In this technique, NAT routers can act as transport- or application-layer traffic relays, allowing clients to establish or receive connections over them. Hence, the NAT router acts as a *proxy* for the traffic.

*SOCKS* [72] is a standard transport-layer gateway that allows hosts to initiate and receive TCP and UDP connections. A host, for example, can connect to the SOCKS server in the NAT router, and request it to initiate a TCP connection to an external server. Similar to the port-forwarding protocols, the host can also request to bind a port to its address, so that it can receive incoming connections.

*Application-Level Gateway* (ALG), is defined in RFC2663 [73] as an application-specific agent that runs on the NAT router and allows an application on the host to establish and receive connections. It allows NAT to perform address and port translation for certain applications, such as File Transfer Protocol (FTP), Session Initiation Protocol (SIP), and Real Time Streaming Protocol (RTSP).

All control-based NAT traversal techniques allow clients to map a port to receive incoming connections through it. Traffic *gateway* techniques are different from *port-forwarding* techniques only in the way they handle outgoing connections. Gateway techniques relay the traffic over the NAT router rather than translating it normally like port-forwarding techniques do.

## **Behavior-Based NAT Traversal**

In behavior-based NAT traversal techniques, the NAT router does not provide any additional functionality to help hosts establish connections. Instead, hosts exploit the behavior of NAT in order to achieve connectivity. Most existing behavior-based NAT traversal protocols are based on one of three fundamental techniques: *connection reversal*, *hole-punching*, and *relaying* [74].

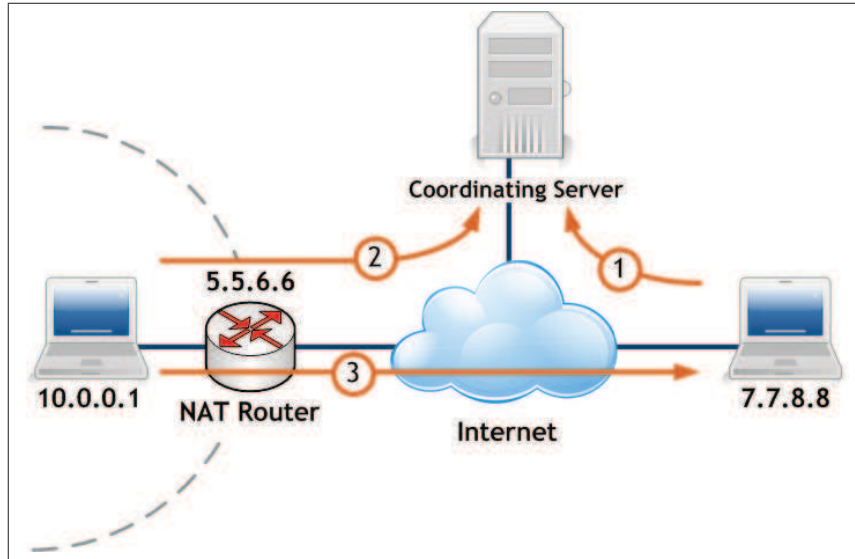


Figure 5.8: Connection reversal technique is used when one of the peers is behind NAT and the other one is not

*Connection reversal* is a straightforward but limited technique that is used when one of the peers is behind NAT and the other is not. First, The two peers communicate through a coordinating server to exchange their connection status. The server determines which peer is behind NAT, and informs it to initiate the connection, even if it is acting as a server in the p2p application, as shown in figure 5.8. This way, the connection will pass through the NAT router normally, since it is initiated from within the NAT to an external host.

The connection reversal technique is only useful when one of the two hosts is not behind NAT, i.e., it has a public IP address. It does not work in the situation where the two hosts are behind NAT.

Connection reversal is used in some peer-to-peer applications, such as Kazaa [75] and Limewire [76]. It helps in establishing connection when only one of the peers is behind NAT, but not when both peers are behind NAT.

*Hole-punching* is a technique that enables two peers, who are both behind NAT, to setup a direct p2p session with the help of a coordinating server. Hole-punching is based on the idea that NAT routers use their NAT tables to map responses of outgoing packets. Each peer, with the help of the coordinating server, initiates a connection to the other peer. The NAT routers will add entries in their tables to map responses coming to the same source ports. The initial packets may be dropped at the other peer's NAT because the NAT table entry is not added yet. Once both NAT routers are *hole-punched*, the two peers can communicate directly.

Hole-punching, as shown in figure 5.9, assumes that the two peers, *A* and *B*, already have active sessions with the coordinating server. The server is used to inform each peer about the public IP address and port of the other peer. Peer *A* sends a packet to *B*'s public IP address. That packet is dropped by *B*'s NAT, because there is no NAT table entry that maps incoming traffic on that port yet. However, an entry is added to *A*'s NAT table, allowing incoming packets on that port to be forwarded to peer *A*. Peer *B* now sends a packet, which is received by peer *A*. At the same time, *B*'s NAT router adds an entry to its NAT table, allowing *B* to receive packets. Now both *A* and *B* can communicate directly without the need for the server, because both NATs accept incoming packets on the mapped ports.

TCP hole-punching is more difficult than UDP, due to the three-way handshake required by TCP [77]. NAT routers check the TCP packet flags, such as

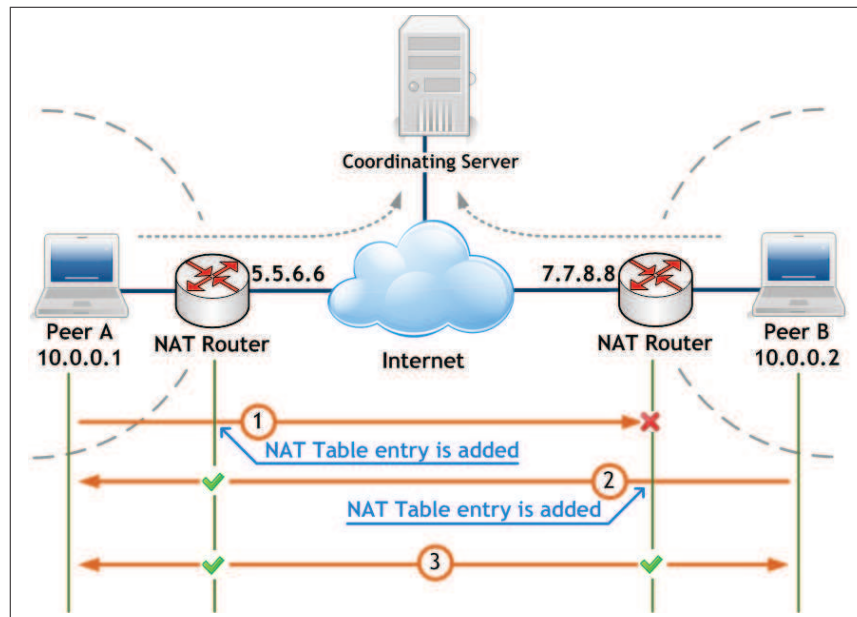


Figure 5.9: Hole-punching technique is used when both peers are behind NAT. Each peer sends a packet to add an entry to its NAT table

SYN and ACK, to determine the start of the session, and add the NAT table entry based on these flags. Some techniques, such as IP address spoofing and delayed-packets, are sometimes needed to accomplish the TCP hole-punching process.

*Relaying* is the most reliable, but least efficient, method of p2p communication across NAT. It is the process of carrying all traffic between the two peers over a central host, the relay server, which has a public IP address. Relaying makes the communication look to the NAT router as standard client-server communication, because both peers initiate connections to the relay server. Relaying always works as long as both clients can connect to the relay server. Figure 5.10 shows how peers can use a public relay server to communicate.

Relaying has many disadvantages. It consumes the server's processing power, requires more network bandwidth, and increases the communication latency be-

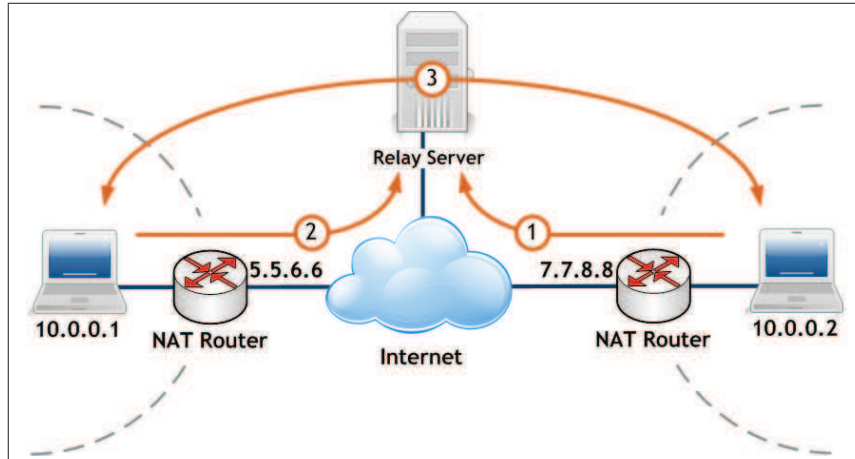


Figure 5.10: Traffic relaying technique works with all NAT configurations, as it makes peers act as client to the relay server

tween the peering clients. Moreover, relaying eliminates the advantages of using peer-to-peer systems, as the network becomes centralized at the relay server. Nevertheless, since there is no more efficient technique that works reliably on all existing NATs, relaying is a useful fallback strategy when no other NAT traversal technique works.

Many behavior-based NAT Traversal protocols have been proposed. *Simple Traversal of UDP Through NAT* (STUN) [64] was introduced as a complete NAT traversal solution based on hole-punching. It allowed clients to determine their NAT type, and perform hole-punching with the help of a STUN server. The protocol was proven to be faulty, and was modified later to become a tool for NAT information discovery, instead of a NAT traversal solution. Its name has been changed to *Session Traversal Utilities for NAT* [78]. STUN is now used as a knowledge-collection tool in many other NAT traversal protocols.

A modification of STUN to support TCP was proposed by Guha in [79]. The



protocol, *Simple traversal of UDP through NATs and TCP* (STUNT), uses more complex operations, such as sending packets with low Time-To-Live (TTL), IP address spoofing, and modifying TCP sequence numbers.

Wacker et al. [80] proposed a method of using STUN in peer-to-peer networks. In this method, the peers who are not behind NAT, having public IP address, are used as STUN servers and relay servers to help other peers who are behind NAT. The method classifies the NAT type of both peers, then selects whether to use connection reversal, hole-punching, or relaying to establish the connectivity.

NATBLASTER [81] is another TCP hole-punching protocol. It is similar to STUNT, but uses port-prediction to determine which TCP ports will be used next by the NAT router based on its behavior.

*Traversal Using Relays around NAT* (TURN) [82] is a relaying protocol that is designed as an extension for STUN. It uses relay servers to transfer UDP traffic between hosts.

*Interactive Connectivity Establishment* (ICE) [83] is a comprehensive protocol for NAT traversal that uses STUN, TURN, and Session Initiation Protocol (SIP) to establish UDP connections between peers. It runs a set of tests with the help of STUN servers in order to determine which technique to use. ICE-TCP [84] is an extension that supports the establishment of TCP connections with ICE.

Guha proposed a NAT traversal protocol called *NUTSS* (**N**AT, **U**RI, **T**unnel, **S**IP, and **S**TUN) [85]. It uses different techniques to establish TCP and UDP connections, such as SIP, STUN, and port prediction. However, this protocol requires

some packets to be sent with spoofed IP addresses, which makes it impractical for Internet use.

Eppinger [86] proposed a software approach that utilizes hole-punching to initiate TCP connections between peers. This approach works only with cone NATs, but not with symmetric NATs.

A solution for symmetric NAT traversal is proposed by Wei et al. [87]. It is based on *multi hole-punching*, where each host sends a large number of packets with different port numbers in order to add more entries in the NAT table. Then, port-prediction is used to determine the mapped ports which can accept incoming traffic. This solution shows high success rate of connectivity, where 97% of the tested NAT were able to establish UDP connections.

Chen et al. [88] have proposed NAT-next-generation (NATng), a modification of DNS and NAT to support the use of public and private IP addresses together. Hosts can initiate connections to other hosts behind NATng directly by addressing them using the tuple of the public IP address and the private IP address. This technique, however, requires the modification of NAT and DNS to support the use of two IP addresses together.

Skype, a popular p2p voice-over-IP (VoIP) application, uses a set of NAT traversal techniques. It uses UDP hole-punching to traverse cone NAT [89]. When hole-punching is not working, as the case with symmetric NAT, Skype relays traffic over other peers on the network who have public IP addresses [90].

## 5.3 Qualitative Analysis of NAT Traversal Techniques

Most of the reviewed NAT traversal techniques and protocols are designed to work with single-level NAT, i.e., each host is behind one NAT router that has a public IP address. As shown earlier, the NAT-based solution for Internet denial can result in having two layers of NAT for the local peers, as they may already have NAT in their local networks. Only few research works address the multi-level NAT problem.

For each NAT traversal technique, the connectivity and performance issues will be discussed, and both single-level NAT and multi-level NAT situations are considered.

### 5.3.1 Analysis of Control-Based Techniques

In normal NAT networks, control-based techniques are designed to work with all types of NAT, as the NAT router provides the additional functionality that helps hosts establish connections. Both port-forwarding and gateway are expected to work when each peer is either directly connected or behind a single NAT router.

However, when another layer of NAT is introduced, as shown in figure 5.11, the connectivity is not always possible unless clients are aware of the existence of two NATs. The NAT router 3.3.4.4 represents the one added for the Internet denial solution, and the other two NAT routers, 5.5.6.6 and 8.8.9.9, are the local and remote network NAT routers, respectively. We notice that on the local side,

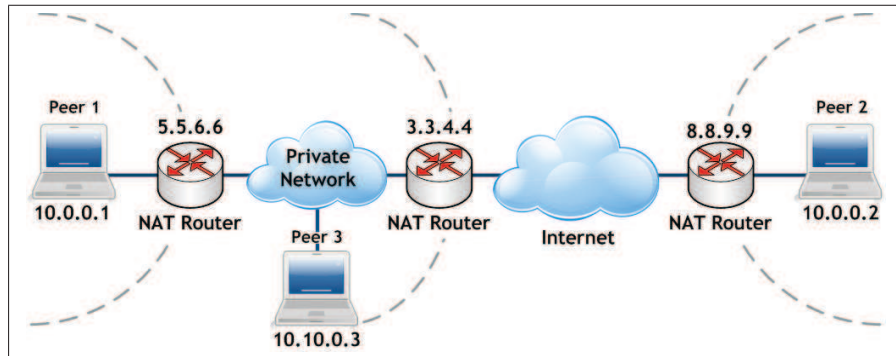


Figure 5.11: Multi-level NAT example. Local peer is behind two levels of NAT, and remote peer is behind one NAT router.

peers can either be behind two levels of NAT, or can be directly connected to the second NAT router, 3.3.4.4.

### Connectivity of Port-Forwarding Technique

When peer 2 uses port-forwarding for NAT traversal, its NAT router will be able to forward all incoming traffic to that peer. Whether the connection is initiated from peer 1 or peer 3, the connection will always be established.

On the other hand, when peer 3 is using port-forwarding to receive connections, it will also be accessible to peer 1 and peer 2. A connection initiated from peer 1 to peer 3 will pass through the NAT router 5.5.6.6 as an outgoing connection. Therefore, the connection will be established. However, peer 3 needs to use port-forwarding on the NAT router 3.3.4.4 in order to be able to receive connections from peer 2 or other hosts on the Internet.

Finally, peer 1 can use port-forwarding on its local router, 5.5.6.6, to receive connections from the private network and peer 3. However, peer 1 will

not be able to receive any connection from peer 2 unless it uses port-forwarding on the two NAT routers, 5.5.6.6 and 3.3.4.4. Therefore, peer 1 needs to perform two port mappings in order to be able to receive connections from Internet hosts. This requires modifications of existing port-forwarding protocols in the client in order to support multi-level NAT mappings.

### **Connectivity of Traffic Gateway Technique**

In traffic gateway NAT traversal, peers use the NAT router for both establishing and receiving connections. When peer 1 initiates a connection to peer 3, it uses its local NAT router, 5.5.6.6, as a gateway. However, to initiate a connection to peer 2, it must use the external NAT router as the gateway. Therefore, peers behind two-levels of NAT must be able to determine which NAT router should be used as a traffic gateway. A peer can try to perform the connection using both NATs, and use the one that connects successfully. However, it should be aware of the presence of the two NAT routers.

When peer 2 and peer 3 try to establish connections, they use their gateway NAT router as the traffic gateway. The destination must also use the traffic gateway technique to accept incoming connections through NAT.

### **Performance of Control-Based Techniques**

Control-based techniques, namely port-forwarding and traffic gateway, cooperate with the NAT router to perform NAT traversal. The performance impact of these techniques is limited to the setup overhead, where peers communicate with

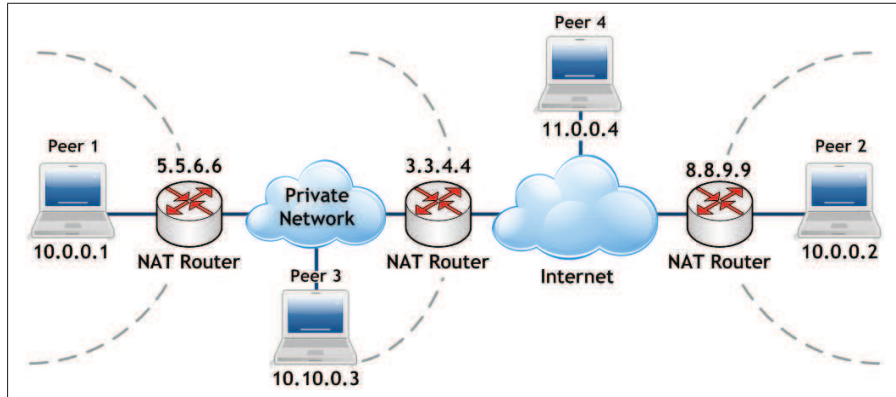


Figure 5.12: Multi-level NAT example. Local peers is behind one or two levels of NAT, and remote peers are directly connected or behind one NAT router.

the NAT router to map a specific port or socket. The remaining packets are not affected by the use of port-forwarding or traffic gateway. Therefore, control-based techniques have very small impact on the performance which can be negligible when the connection session time is relatively long.

### 5.3.2 Analysis of Behavior-Based Techniques

Behavior-based techniques, namely connection reversal, relaying, and hole-punching, can be affected by the introduction of a second layer of NAT in terms of connectivity and performance. Figure 5.12 shows an example of a situation where two levels of NAT are on the local side, and one NAT router is on the remote side. Hosts can be either directly connected to the Internet, such as peer 4; connected to one NAT router, such as peers 2 and 3, or connected to two NAT routers, such as peer 1.

## Connection Reversal

Connection reversal does not add any overhead to the connection process. However, it only works in situations where one of the peers has a public IP address. In the example shown in figure 5.12, peer 4 can be reached by all other peers using connection reversal, since it has a public IP address. Connection reversal does not provide any means for connectivity between other peers. However, we notice that peer 1 and 3 can communicate using connection reversal, because peer 3 is on the *external* network of peer 1.

There is no performance impact when connection reversal is used. However, it does not work with most scenarios, and can only be considered as a limited NAT traversal technique.

## UDP and TCP Hole-Punching

Hole-punching depends heavily on the behavior of NAT. STUN [64] and ICE [83] are examples of protocols that use a set of tests to determine the behavior of NAT before they perform hole-punching. We assume that the NAT routers in our example are cone-type that allow hole-punching.

Ford et al. [91] have studied the situation of two-level NAT. They have shown that TCP and UDP hole-punching can be used to communicate between hosts in different NATs. However, that depends heavily on the types of NATs that hosts are connected to. 82% of the NAT routers supported UDP hole-punching, but only 64% supported TCP hole-punching.

In the previous example shown in figure 5.12, when peer 1 and peer 2 use hole-punching to establish a connection, each peer sends an initial packet to setup a NAT table entry in its NAT router. The packet sent by peer 1 will go through both NAT routers, 5.5.6.6 and 3.3.4.4. Each router will add a NAT table entry for that packet to map incoming responses. Similarly, the packet sent by peer 2 will add an entry in the NAT router 8.8.9.9. At this point, all routers can receive traffic on the mapped ports, and the connection is now established.

Similarly, connectivity between peer 2 and peer 3 can also be established using hole-punching. Each initial packet will go through one NAT router, and a NAT table entry is added at each router. Connections to peer 4, on the other hand, do not require hole-punching, as that peer has a public IP address. Moreover, peer 1 can connect directly to peer 3 without the need for hole-punching, as there is only one NAT router between them.

The complications of hole-punching appear when one of the NAT is a symmetric NAT. The mapped external port are usually undetermined, and therefore, the other peer does not know which port should be the destination of its packet. Some research work has been done on the prediction of mapped ports [81, 87].

In terms of performance, hole-punching requires sending a number of initial packets to map NAT table entries. Once the connection is established, however, there is no impact on the performance of the network. Hence, the setup overhead is the only impact on the network performance, and it is negligibly small for normal-length sessions.



## Relaying

Relaying is the only NAT traversal technique that works with all NAT types and topologies. In the previous example, if peer 4 is a relay server, all other peers can connect to it, as it has a public IP address. Traffic, therefore, can be exchanged between peers by relaying through peer 4.

However, the performance impact of relaying is very high. For example, when peer 1 and peer 2 are communicating through peer 4 as a relay, each peer sends packets to peer 4, and then peer 4 sends them to the other peer. The relay server uses twice the bandwidth of the traffic that is exchanged between peers, because it receives and sends traffic to both peers. This results in high computational requirements of the relay.

End-to-end delay is also affected by the introduction of a relay. Each packet is sent first to the relay server, then the relay sends it to the other peer. Therefore, end-to-end delay may double. For high-traffic scenarios, it is expected that end-to-end delay increases even more due to the queuing that takes place in the relay server and the intermediate routers.

## 5.4 Usage of NAT Traversal for the Internet Denial Solution

The proposed solution for Internet denial is based on using NAT to hide the traffic identity. Therefore, p2p applications need to use NAT traversal protocols

in order to achieve connectivity. The modification of these applications may take place on the local clients, the NAT routers, and/or the remote clients on the Internet, depending on the NAT traversal protocol that is used.

#### **5.4.1 Deployment of Control-based NAT Traversal**

Control-based protocols require NAT routers to provide the traversal service in order to allow clients to map ports to their addresses. Therefore, the NAT router should support one or more of the control-based protocols. Moreover, the p2p application that clients are using should also support the usage of these protocols. The presence of two NAT routers may require additional modifications of the p2p application.

Although the control-based techniques provide high connectivity compared with behavior-based techniques, they have some security issues. Most control-based protocols are designed for small networks, such as home networks. They do not fit large-scale network. When the network is larger, the port-forwarding could be abused, where a client may map many ports to its address. Moreover, when there are many clients, many port-forwarding requests will fail because other hosts have already allocated the selected ports. Therefore, port-forwarding protocols are not viable for large NAT environments.

### 5.4.2 Deployment of Behavior-based NAT Traversal

Behavior-based NAT traversal protocols are more suitable for large-scale NAT networks. As shown earlier, the three fundamental techniques that all protocols are using are connection reversal, hole-punching, and relaying. All the three techniques require modifying the p2p applications in both local and remote peers. However, no modification is required in the NAT routers.

Connection reversal, as shown earlier, is limited to scenarios where one of the peers has a public IP address. On the other hand, hole-punching and relaying can work with different scenarios. Most popular p2p applications, such as Skype, BitTorrent, Gnutella, and LimeWire, support NAT traversal protocols that support one or both of these techniques [92].

When the Internet denial solution is deployed, p2p applications in the private network can use hole-punching to establish connections. However, some NAT environments do not allow or support hole-punching. Therefore, relaying is used as the last solution when other techniques are unusable.

## 5.5 Performance Evaluation of NAT Traversal

### Using Relaying

The impact of most NAT traversal techniques is limited to the setup time. Relaying, on the other hand, has much more impact on the network performance. A relay server is required to have enough bandwidth and processing power to

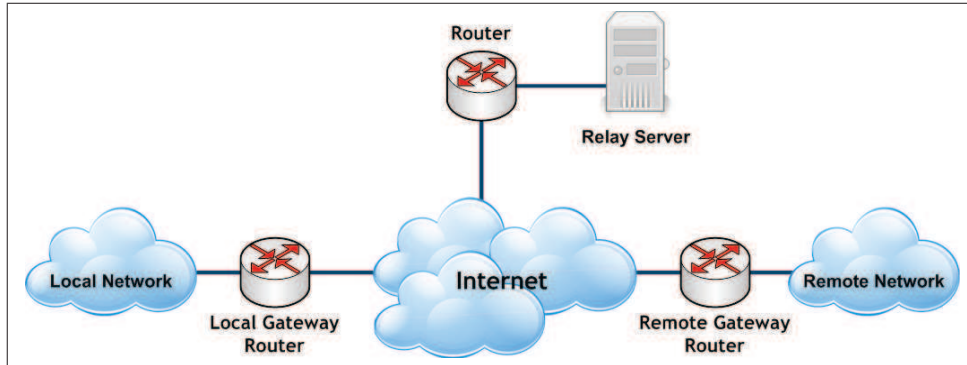


Figure 5.13: Simulated scenario to measure the effect of relaying in peer-to-peer communication

carry traffic between peers. Therefore, the performance of relaying is evaluated through simulations in order to measure the cost of this solution.

The objective of the simulations is to compare the network performance before and after adding a relay server to carry traffic between peers. Two scenarios are compared: direct communication between peers, where the relay server is not used; and the communication through relaying server, where all traffic between peers is carried by an intermediate relay server. The used performance metrics are the end-to-end delay of packets, and the traffic throughput.

### 5.5.1 Simulation Scenario and Setup

Figure 5.13 shows the network used in the simulations. There are two networks, local and remote, that are connected through gateway routers to the Internet cloud. A third network, consisting of the relay server, is also connected through a router to the Internet cloud.

The local and remote networks are 100BaseT *Fast Ethernet* networks. Each

network has 10 hosts that will serve as peers in the p2p network. All the routers are based on the generic router model in OPNET. The two routers connecting LANs to the Internet are connected to the central Internet cloud using DS-1 links, which provide a data rate of 1.544 Mbps. The router that connects the relay server to the Internet is connected through a DS-3 link, providing a data rate of 44.736 Mbps.

The link that connects the relay server to the Internet has higher bandwidth than the other links connecting peers to the Internet. The reason is that the relay server requires twice the bandwidth that is required by peers as it carries traffic over both networks.

The simulated application is Video Conferencing, which runs over UDP. Four traffic scenarios are simulated: 300 Kbps, 600 Kbps, 900 Kbps, and 1,200 Kbps. These traffic values represent the amount of traffic sent from peers on one network to peers on the other network.

Each simulation is run with 5 different seed values, and the average of the 5 results is computed. Performance is evaluated by comparing the scenario of direct communication between peers with the scenario of using the relay to carry traffic.

### **5.5.2 Simulation of End-to-End Delay**

The effect of relaying on peer-to-peer network is expected to be high. The reason is the relay carries twice the packets; traffic from each peer to the relay server, and traffic from the relay to each peer. Therefore, an increase by 100% is

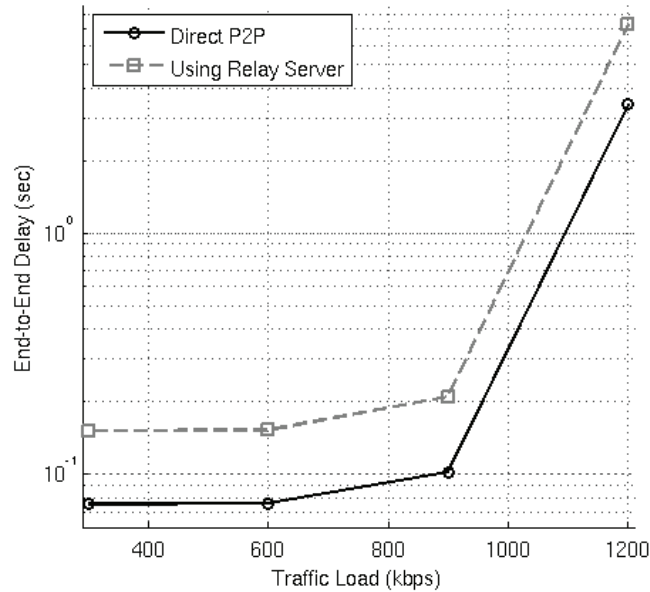


Figure 5.14: End-to-end delay for P2P network (logarithmic scale)

expected to take place when relaying is used.

The measured end-to-end delay for the direct and relaying scenarios is shown in figure 5.14. We notice a large increase in the delay when relaying is used. Moreover, higher traffic adds more delay on packets, as noticed when the traffic reaches 1,200 kbps. This increase in the delay is due to queuing that takes place in the routers and the relay server.

Figure 5.15 shows the increase of end-to-end delay due to the use of relay. The increase is computed as  $(Delay_{Relay} - Delay_{Direct})$ . It is seen that the absolute increase in the delay is higher when traffic is high. Furthermore, looking at the relative increase of the end-to-end delay, computed as  $\frac{(Delay_{Relay} - Delay_{Direct})}{Delay_{Direct}}$  and shown in figure 5.16, we see that relaying doubles the delay. This is the expected behavior as explained earlier. Moreover, higher traffic causes a larger increase in

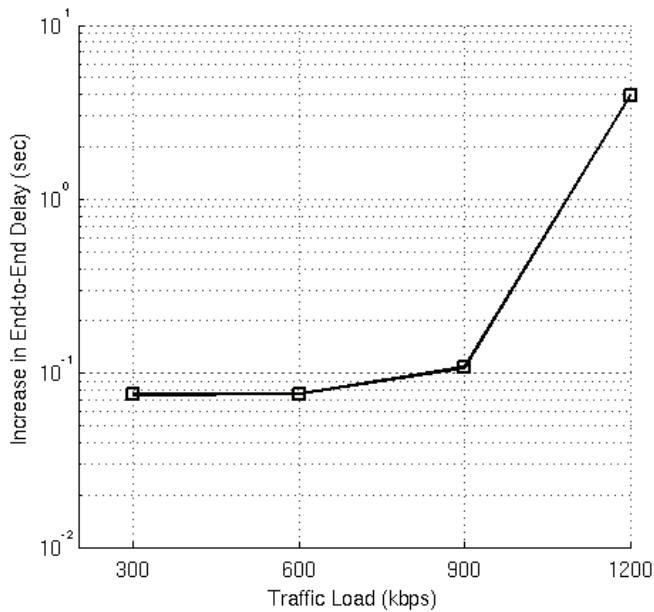


Figure 5.15: Increase in End-to-end delay for P2P network (logarithmic scale)

the delay, as the relative increase reaches 115% when the traffic is 1,200 kbps. The reason is that the transmission queue at the relay server is occupied by the traffic of both sides at the same time, since the relay carries traffic for both peers. This causes higher queuing delay, which increases the end-to-end delay as shown in the figure.

### 5.5.3 Simulation of Traffic Bandwidth

Relaying requires high bandwidth because the relay server carries the traffic for both directions. Therefore, it is expected that the amount of traffic transmitted on the relay's link is double the one transmitted by a peer. The relaying scenario is simulated, and the traffic throughput is measured at the relay's link that connects it to the Internet, and at the peer's link that connects the LAN router to the

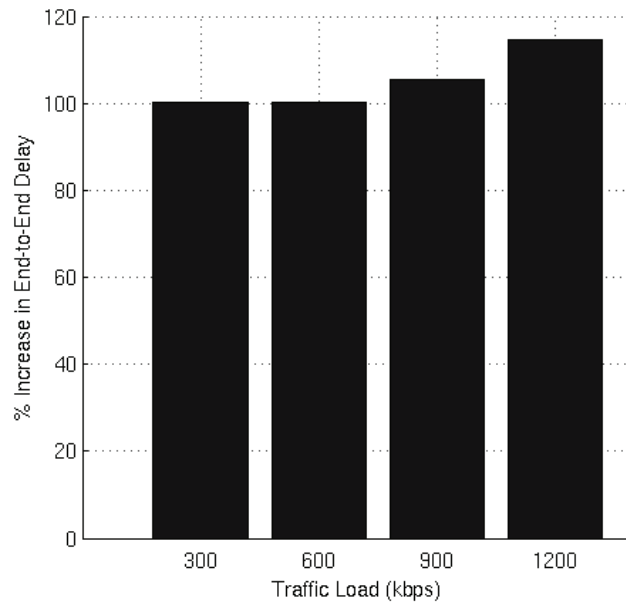


Figure 5.16: Relative increase in End-to-end delay for P2P network

Internet.

Figure 5.17 shows the amount of measured traffic at both links, the peer's and the relay server's. It is noticed that the relay transmits higher traffic than the peers, since it is carrying the traffic in both directions. The relative difference between the throughput of relay and peer is shown in figure 5.18. All scenarios experience an increase of about 100%, which is expected as the relay carries double the traffic.

#### 5.5.4 Conclusion

The simulation results show that end-to-end delay is highly affected by the use of a relay. End-to-end delay suffers an increase of more than 100%, specially when the traffic is high. In terms of bandwidth, the relay server requires at least



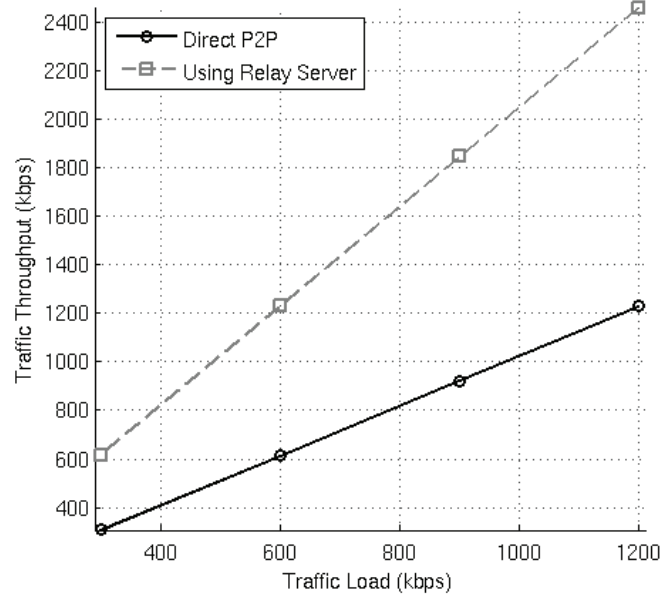


Figure 5.17: Throughput of P2P network with relaying

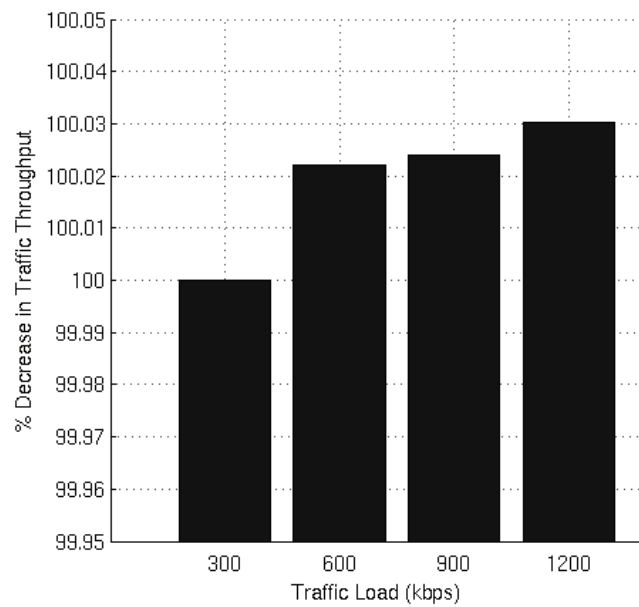


Figure 5.18: Relative difference between relay's throughput and peer's throughput

twice the bandwidth, since the relay server carries twice the amount of traffic sent by peers.

# CHAPTER 6

## DISCUSSION AND RECOMMENDATIONS

In this chapter, the complete solution for Internet denial is discussed, together with some design, implementation and performance considerations and recommendations. An example scenario of Internet denial in Saudi Arabia is also discussed to illustrate how the solution can be applied.

### 6.1 Overview of the NAT-Based Solution

#### 6.1.1 Comparison of Different Solutions for Internet Denial

In chapter 2.3.4, a number of solutions for Internet denial were introduced. The proposed NAT solution has the lowest overhead in terms of performance and required modifications of the network. BGP Tuning techniques require coopera-

tion of other ISPs, and do not provide a way of sending traffic to hosts within the malicious ISP. Tunneling and anonymous routing introduce large overhead that impacts the network performance. They do not allow contacting hosts within the malicious ISP.

The NAT solution has very low overhead, which takes place as extra processing delay at the NAT router. NAT also allows the hosts to contact all other hosts on the Internet, including the ones within the malicious network. Hence, the NAT solution can be considered as one of the most efficient solutions in terms of performance and connectivity.

### **6.1.2 Design Recommendations**

NAT translates all private IP addresses into a smaller set of public IP addresses. It could translate traffic from a large network into a single IP address. However, more IP addresses are needed to overcome limitations such as the number of mappable ports.

The proposed NAT solution does not require extra resources other than the NAT routers at the gateway level. Bandwidth, for example, is hardly affected when NAT is enabled. Therefore, the existing network bandwidth should be sufficient when the solution is deployed. Nevertheless, the design can be scalable, as discussed earlier in section 3.3.4. These scalability designs can be used by the entire blocked network, or by subnetworks of it.

Similarly, the proposed solutions for servers can be scaled for larger networks.

In the proposed HTTP solution, two methods for scalability and load-balancing are presented earlier: using multiple web switches, and using DNS for load balancing. Therefore, the NAT routers are not the bottleneck of incoming traffic anymore, as more routers and web switches can be added to handle a larger amount of requests.

### 6.1.3 Performance Considerations

In terms of performance, the NAT solution has a lower performance impact than other solutions. The maximum effect measured through the simulations does not exceed an increase of 0.4% for the end-to-end delay, and a decrease of 0.3% for the traffic throughput, except for the p2p applications when relaying is used. Moreover, these values are measured for very high NAT delay values. Actual NAT routers can perform faster processing than the simulated one. Therefore, the effect of NAT in real networks is negligibly small.

Similarly, the proposed solutions for HTTP and SMTP servers have also low impact on the network. The solution for HTTP servers adds a small overhead to the first packet of each request. The remaining packets only experience NAT-like processing delay. Therefore, the results of simulating this solution were very close to the ones that were measured in the NAT simulation. The SMTP solution, on the other hand, adds an intermediate layer of email relaying. The delivery of email messages are not as time-critical as other applications. The time for a message to be delivered from the external server to the email relay is not changed. However,

the message must be sent again from the relay to the actual server. This may add some delay for the end-to-end delivery. Nevertheless, this delay is not very critical for an application like SMTP.

Peer-to-peer connectivity is highly affected by NAT. Peers who used to have direct connections are now forced to use NAT traversal protocols in order to establish connections. As discussed earlier, the performance overhead of most NAT traversal protocols is limited to the setup time. However, relaying is the only technique that has a large effect on performance. Simulations have shown that the relay server requires double the bandwidth that peers use in order to be able to deliver traffic between them. End-to-end connectivity is also increased by at least 100% when a relay server is used. The impact of relaying is very high, but as it is the only NAT traversal solution that would work with all possible NAT types and topologies, some peers may have to use it regardless of its performance issues.

#### **6.1.4 Solution Limitations**

The NAT solution for Internet denial has some limitations in terms of connectivity and durability. NAT limits end-to-end connectivity, causing services and some p2p applications to be inaccessible. We have proposed solutions for Web and Email servers, but other services within the private network are still inaccessible.

Peer-to-peer applications that do not support NAT traversal protocols may

not work correctly behind NAT. Moreover, the Internet denial solution adds a second layer of NAT. Hence, even if the applications support NAT traversal, some of them may only work with single-NAT scenarios, but not multi-NAT.

The proposed solution uses identity hiding to bypass the Malicious ISP that is blocking the traffic. The IP addresses that are used in the NAT solution may be borrowed from another network. If the malicious ISP detected that these IP addresses are being used by the blocked network, it would block them too. This would require replacing the IP addresses with new ones. Therefore, the NAT solution would only work if the IP addresses are kept hidden from the malicious ISP. This is possible when the private network consists of only clients, but no servers. The provided solutions for servers behind NAT expose the new IP addresses in the DNS, allowing the malicious ISP to detect that these IP addresses are being used by the blocked network, and to block them eventually.

## **6.2 Case Study: Internet Denial in Saudi Arabia**

In this section, we will look into a hypothetical scenario where a number of higher-tier ISPs perform Internet denial on the network in Saudi Arabia. The design and deployment issues of implementing the proposed solution are discussed.

### **6.2.1 Internet Structure in Saudi Arabia**

Saudi Arabia is connected to the Internet through a number of international ISPs. The network inside Saudi Arabia consists of two layers of ISPs: Data

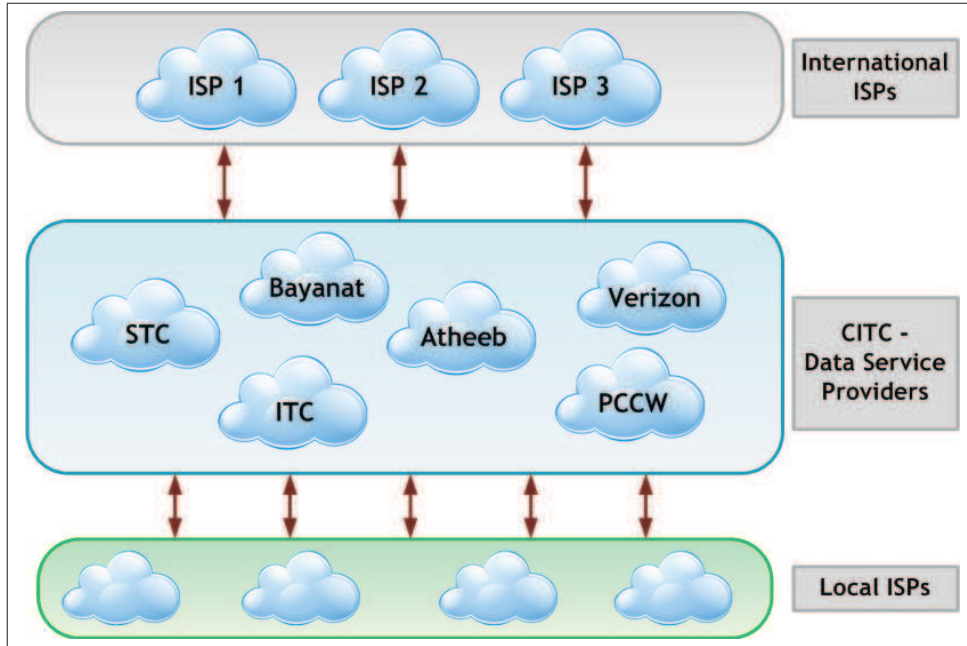


Figure 6.1: Internet structure in Saudi Arabia

service providers (DSPs), and local Internet service providers (local ISPs). DSPs are connected directly to the international ISPs through a number of physical links. On the other hand, local ISPs provide Internet services to end users, and are connected to one or more DSPs to access the Internet. The structure of Internet in Saudi Arabia is shown in figure 6.1.

### 6.2.2 Internet Denial

The Internet denial scenario that we assume is performed by one or more international high-tier malicious ISPs. The malicious ISPs could be connected directly to the Saudi DSPs, or they could be part of some AS paths that carry the traffic to the destinations.

We assume that the malicious ISPs will block the IP addresses that belong to



all service providers in Saudi Arabia. Hence, the traffic sent from any client in Saudi Arabia will be dropped when it reaches the malicious ISP.

### **6.2.3 Deployment of NAT Solution in Saudi Arabia**

In order to implement the Internet denial solution, one or more non-blocked IP addresses must be available in order to use them as public IP addresses for the NAT routers. These IP addresses could be bought from a neighboring network or country.

NAT routers can be set at the gateway level, i.e., at the links that connect DSPs to the International ISPs. Because NAT does not have any significant impact on the performance, additional links or resources are not needed. The NAT routers could be deployed at the existing links without the need to increase the bandwidth or to add hardware.

Scalability issues, as discussed earlier, can be resolved by using pools of public IP addresses at the routers, and increasing the number of NAT routers that are connected. These solutions would resolve any limitations that are caused by the lack of memory, bandwidth, or processing power.

The IP addresses within the network should not be changed. The addressing system should be kept the same. This has two advantages: it prevents any address conflicts with the NAT routers that already exist in the network, and it keeps the transition to the NAT solution transparent from the end users. NAT routers should be configured to treat the IP address blocks within the network as private

IP addresses.

Users now can access the Internet through the NAT routers as clients, and the malicious ISPs would route the packets normally because of the hidden identity.

In order to allow clients on the Internet to access services within Saudi Arabia, the proposed solutions for servers should be implemented as well. The solution for HTTP servers requires the addition of web switches in order to handle incoming requests. In the earlier discussions of the solution's design in chapter 3.5, we have seen that web switches can be separated from the NAT routers. Hence, incoming connections are separated from outgoing ones.

Similarly, the solution for SMTP servers is needed to allow external email servers to deliver messages to the SMTP servers within Saudi Arabia. The solution requires the use of one or more SMTP relay servers that are used as the mail exchanges for all mail services within Saudi Arabia.

It is important to separate the used IP addresses for the servers and the clients. In order to achieve higher durability of the solution, the IP addresses that are used for the servers should be different than the ones that are used for the clients outgoing connections. This decreases the probability that the malicious ISPs will detect that the blocked network is using different IP addresses.

The domain names for all the servers within Saudi Arabia should be updated in the external DNS servers with the new public IP addresses. Internal DNS servers, on the other hand, do not require any changes as the internal structure and addressing of the network are not changed.

Finally, p2p applications that are used by clients within Saudi Arabia must support one or more NAT traversal protocols in order to work correctly. The earlier analysis of NAT traversal protocols have shown that most techniques can actually work with the dual-NAT problem, but some of them require modifications in order to handle this scenario. Therefore, few p2p applications may not work correctly with this solution. However, most existing p2p applications implement NAT traversal protocols, such as hole-punching and relaying, that would work properly with this scenario.

## CHAPTER 7

# CONCLUSION

This chapter concludes the thesis by outlining the contributions achieved. Moreover, it discusses some of the open problems for future work.

### 7.1 Summary of the Contributions

In this thesis, the following has been achieved:

1. A NAT-based solution to the *Internet denial* problem is designed and evaluated.
2. Techniques for HTTP and SMTP servers reachability behind NAT are proposed and evaluated.
3. Different NAT traversal techniques for peer-to-peer applications are evaluated in terms of performance and connectivity.

## 7.2 Future Work

Many issues in the *Internet denial* problem are open for further research. Some of the issues are:

- **Detection of Internet denial:** The detection process is important to find out whether traffic *blackholing* is taking place, which ISP is performing it, and whether it is for all generated traffic, or only for a specific type of packets. The challenge of this process is to distinguish whether the dropping of packets is caused by routing-level Internet denial; by other malicious causes, such as Denial of Service (DoS) attacks; or by non-malicious reasons, like network congestion, server unavailability, or connection time-outs.
- **Server reachability behind NAT:** In this thesis, we resolved the servers-behind-NAT problem for Web and E-Mail servers. Solutions for other types of services, such as FTP, VPN, Telnet, etc. are needed.
- **Subsequent Internet denial of ISPs:** We assumed that the malicious ISP will block the original set of IP addresses, and it will not block any subsequently used IP addresses. This assumption depends heavily on the behavior and motivation of the malicious ISP, as it may put some effort in tracking the new IP addresses and block them. The proposed solution may not withstand such actions. Therefore, more investigation is needed in this area.
- **P2P connectivity behind NAT:** The qualitative study of the existing

NAT traversal techniques and protocols shows that many of these techniques do not function correctly with multi-NAT scenarios. Some of these techniques require modifications in order to achieve connectivity, while other simply do not work with such scenarios. Better NAT traversal solutions are needed to enable p2p connectivity behind multi-level NAT.

# References

- [1] “Growing business dependence on the Internet: New risks require CEO action,” report, Business Roundtable, Sept. 2007.
- [2] A. A. Zain, “Cable damage hits one million internet users in UAE,” *Khaleej Times*, Feb. 2008.
- [3] H. Timmons, “Undersea cables cut, disrupting Net,” *The New York Times*, Mar. 2008.
- [4] “2-breaks in cables to Middle East disrupt Internet,” *Reuters*, Dec. 2009.
- [5] J. paul Kamath, “Indian internet disruption hits UK businesses,” *ComputerWeekly*, Jan. 2008.
- [6] “Router glitch cuts Net access,” *CNET News*, Apr. 1997.
- [7] J. Raphael, “Google’s Gmail outage: A familiar feeling,” *PC World*, Sept. 2009.
- [8] “Microsoft reports worldwide Hotmail outage,” *CBC News*, Feb. 2008.

- [9] P. Roberts, “Comcast suffers DNS outage, denies pharming link,” *InfoWorld*, Apr. 2005.
- [10] N. Mook, “DNS outage takes down Google,” *Betanews*, May 2005.
- [11] J. Hawkinson and T. Bates, “Guidelines for creation, selection, and registration of an autonomous system (AS),” RFC 1930, Internet Engineering Task Force, Mar. 1996.
- [12] G. Culpin, “Discovery of internet topology through active probing,” Master’s thesis, Catholic University of Louvain, Aug. 2006.
- [13] T. Bates, P. Smith, and G. Huston, “CIDR report.” <http://www.cidr-report.org/as2.0/>, Mar. 2010.
- [14] “Autonomous system (AS) numbers.” <http://www.iana.org/assignments/as-numbers/>, Sept. 2009.
- [15] K. Butler, T. Farley, P. McDaniel, and J. Rexford, “A survey of BGP security issues and solutions,” *Proceedings of the IEEE*, vol. 98, pp. 100–122, Jan. 2010.
- [16] M. Lad, R. Oliveira, B. Zhang, and L. Zhang, “Understanding resiliency of internet topology against prefix hijack attacks,” in *Proc. of IEEE/IFIP DSN*, 2007.
- [17] Z. Mao, J. Rexford, J. Wang, and R. Katz, “Towards an accurate AS-level traceroute tool,” in *Proceedings of the 2003 conference on Applications, tech-*



- nologies, architectures, and protocols for computer communications*, pp. 365–378, ACM New York, NY, USA, 2003.
- [18] C. Labovitz, R. Malan, and F. Jahanian, “Origins of internet routing instability,” in *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 1999)*, pp. 218–226, Mar. 1999.
- [19] S. Lee, Y. Yu, S. Nelakuditi, Z.-L. Zhang, and C.-N. Chuah
- [20] N. Poolsappasit and I. Ray, “Enhancing internet domain name system availability by building rings of cooperation among cache resolvers,” in *IEEE SMC Information Assurance and Security Workshop*, pp. 317–324, June 2007.
- [21] A. Nucci, S. Bhattacharyya, N. Taft, and C. Diot, “Igp link weight assignment for operational tier-1 backbones,” no. 4, pp. 789–802, 2007.
- [22] R. Ford, M. Bush, and A. Boulatov, “Internet instability and disturbance: goal or menace?,” in *Proceedings of the 2005 workshop on New security paradigms*, pp. 3–8, Sept. 2005.
- [23] J. Zheng, M. Hu, and L. Zhao, “Enhancing internet robustness against malicious flows using active queue management,” in *Second International Conference on Embedded Software and Systems*, pp. 501–506, Dec. 2005.
- [24] F. Guo, J. Chen, and T.-c. Chiueh, “Spoof detection for preventing dos attacks against dns servers,” in *ICDCS '06: Proceedings of the 26th IEEE*

- International Conference on Distributed Computing Systems*, (Washington, DC, USA), p. 37, IEEE Computer Society, 2006.
- [25] M. Lad, R. Oliveira, B. Zhang, and L. Zhang, “Understanding resiliency of internet topology against prefix hijack attacks,” in *In Proc. IEEE/IFIP DSN*, 2007.
- [26] M. Haungs, R. Pandey, and E. Barr, “Handling catastrophic failures in scalable internet applications,” *Applications and the Internet, IEEE/IPSJ International Symposium on*, vol. 0, p. 188, 2004.
- [27] D. Dolev, S. Jamin, O. Mokryn, and Y. Shavitt, “Internet resiliency to attacks and failures under bgp policy routing,” *Comput. Netw.*, vol. 50, no. 16, pp. 3183–3196, 2006.
- [28] “IPv4 address space registry.” <http://www.iana.org/assignments/ipv4-address-space/>, Feb. 2010.
- [29] L. Daigle, “WHOIS protocol specification,” RFC 3912, Internet Engineering Task Force, Sept. 2004.
- [30] IPInfoDB, “Ip address country block generator.” [http://ipinfodb.com/ip\\_country\\_block.php](http://ipinfodb.com/ip_country_block.php).
- [31] D. Drummond, “A new approach to china.” The Official Google Blog, <http://googleblog.blogspot.com/2010/01/new-approach-to-china.html>, Jan. 2010.

- [32] J. Finkle and D. Bartz, “Twitter hacked, attacker claims iran link.”
- [33] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, and S. Uhlig, “Inter-domain traffic engineering with BGP,” *Communications Magazine, IEEE*, vol. 41, no. 5, pp. 128, 122, 2003.
- [34] B. Quoitin and O. Bonaventure, “A cooperative approach to interdomain traffic engineering,” *IN PROCEEDINGS OF EURONGI*, 2005.
- [35] A. Alrefai, “BGP-based solution for international ISP blocking,” dec 2009.
- [36] C. Perkins, “IP encapsulation within IP,” RFC 2003, Internet Engineering Task Force, Oct. 1996.
- [37] R. Atkinson, “Security architecture for the internet protocol,” RFC 1825, Internet Engineering Task Force”, URL= <http://www.rfc-editor.org/rfc/rfc1825.txt>, Aug. 1995.
- [38] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, “Generic routing encapsulation (GRE),” RFC 2784, Internet Engineering Task Force, Mar. 2000.
- [39] P. F. Syverson, M. G. Reed, and D. M. Goldschlag, “Anonymous connections and onion routing,” *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, vol. 16, pp. 482—494, 1998.
- [40] L. Zhuang, F. Zhou, B. Y. Zhao, and A. Rowstron, “Cashmere: resilient anonymous routing,” in *Proceedings of the 2nd conference on Symposium*

- on Networked Systems Design & Implementation - Volume 2*, pp. 301–314, USENIX Association, 2005.
- [41] M. K. Reiter and A. D. Rubin, “Anonymous Web transactions with Crowds,” *Communications of the ACM*, vol. 42, no. 2, pp. 32–48, 1999.
- [42] C. Shields and B. N. Levine, “A protocol for anonymous communication over the internet,” in *Proceedings of the 7th ACM conference on Computer and communications security*, (Athens, Greece), pp. 33–42, 2000.
- [43] J. Liu, J. Kong, X. Hong, and M. Gerla, “Performance evaluation of anonymous routing protocols in MANETs,” in *IEEE Wireless Communications and Networking Conference*, 2006.
- [44] I. van Beijnum, *Running IPv6*. Apress, Nov. 2005.
- [45] K. Egevang and P. Francis, “The IP network address translator (NAT),” RFC 1631, Internet Engineering Task Force, may 1994.
- [46] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. de, and E. Lear, “Address allocation for private internets,” RFC 1918, Internet Engineering Task Force, Feb. 1996.
- [47] P. Srisuresh and B. Ford, “Unintended consequences of NAT deployments with overlapping address space,” RFC 5684, Internet Engineering Task Force, Feb. 2010.
- [48] J. Doyle and J. Carroll, *Routing TCP/IP, Volume II*. Cisco Press, 2005.

- [49] “CISCO IOS Network Address Translation Q&A.” [http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6640/prod\\_qas0900aecd801ba55a.html](http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6640/prod_qas0900aecd801ba55a.html).
- [50] “System architecture overview for the Juniper networks SSG500 line,” Feb. 2009.
- [51] “OPNET Modeler.” <http://www.opnet.com/>.
- [52] “The Network Simulator – ns-2.” <http://www.isi.edu/nsnam/ns/>.
- [53] R. Ramaswamy, N. Weng, and T. Wolf, “Characterizing network processing delay,” in *IEEE GLOBECOM*, pp. 1629–1634, 2004.
- [54] D. Clark, V. Jacobson, J. Romkey, and H. Salwen, “An analysis of TCP processing overhead,” *IEEE Communications Magazine*, vol. 27, no. 6, pp. 23–29, 1989.
- [55] “Intel IXP2800 network processor specification update,” tech. rep., Intel Corporation, Feb. 2005.
- [56] J. Franks, “Two proposals for HTTP/2.0.” <http://www.ics.uci.edu/pub/ietf/http/hypermail/1994q4/0019.html>, Nov. 1994.
- [57] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, “Hypertext transfer protocol – HTTP/1.1,” RFC 2616, Internet Engineering Task Force, June 1999.

- [58] “Apache virtual host documentation.” <http://httpd.apache.org/docs/2.2/vhosts/>.
- [59] “Use host header names to configure multiple web sites in IIS 6.0.” <http://go.microsoft.com/fwlink/?LinkId=36045>.
- [60] V. Cardellini, P. S. Yu, V. Cardellini, V. Cardellini, E. Casalicchio, E. Casalicchio, P. Yu, M. Colajanni, and M. Colajanni, “The state of the art in locally distributed web-server systems,” *ACM Computing Surveys*, vol. 34, no. 34, pp. 263–311, 2001.
- [61] J. Klensin, “Simple mail transfer protocol,” RFC 2821, Internet Engineering Task Force, Apr. 2001.
- [62] H. Schulze and K. Mochalski, “Internet study 2008/2009,” tech. rep., ipoque, 2009.
- [63] M. Meeker and D. Joseph, “The state of the internet,” in *Web 2.0*, 2006.
- [64] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, “STUN - simple traversal of user datagram protocol (UDP) through network address translators (NATs),” RFC 3489, Internet Engineering Task Force, March 2003.
- [65] F. Audet and C. Jennings, “Network address translation (NAT) behavioral requirements for unicast UDP,” RFC 4787, Internet Engineering Task Force, Jan. 2007.

- [66] B. Wang, X. Wen, S. Yong, and Z. Wei, "A novel NAT traversal mechanism in the heterogeneous environment," 2009.
- [67] M. Ohta, "End to end nat," internet draft, Internet Engineering Task Force, July 2009.
- [68] U. Forum, "Internet gateway device (IGD) standardized device control protocol." <http://www.upnp.org/standardizeddcps/igd.asp>, nov 2001.
- [69] S. Cheshire, M. Krochmal, and K. Sekar, "NAT port mapping protocol (NAT-PMP)," internet draft, Apr. 2008.
- [70] M. Borella, D. Grabelsky, J. Lo, and K. Taniguchi, "Realm specific IP: protocol specification," RFC 3103, Internet Engineering Task Force, oct 2001.
- [71] P. Srisuresh, J. Kuthan, J. Rosenberg, A. Molitor, and A. Rayhan, "Middle-box communication architecture and framework," RFC 3303, Internet Engineering Task Force, Aug. 2002.
- [72] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, "SOCKS protocol version 5," RFC 1928, Internet Engineering Task Force, mar 1996.
- [73] P. Srisuresh and M. Holdrege, "Ip network address translator (NAT) terminology and considerations," RFC 2663, Aug. 1999.
- [74] K. Pussep, M. Weinert, N. Liebau, and R. Steinmetz, "Flexible Framework for NAT Traversal in Peer-to-Peer Applications," tech. rep., KOM-TR-2007-

06, KOM-Multimedia Communications Lab, Technische Universit

”at Darmstadt, 2007.

- [75] J. Jessup, “How Kazaa works,” 2003.
- [76] Lime Wire LLC, “Frequently asked questions.”  
<http://www.limewire.com/english/content/faq.shtml>.
- [77] Z. Hu, “NAT Traversal Techniques and Peer-to-Peer Applications,” *Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology*.
- [78] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, “Session traversal utilities for NAT (STUN),” RFC 5389, Internet Engineering Task Force, October 2008.
- [79] S. Guha, “Simple traversal of UDP through NATs and TCP too (STUNT),” internet draft.
- [80] A. Wacker, G. Schiele, S. Holzapfel, and T. Weis, “A NAT traversal mechanism for peer-to-peer networks,”
- [81] A. Biggadike, D. Ferullo, G. Wilson, and A. Perrig, “NATBLASTER: Establishing TCP connections between hosts behind NATs,” in *Proceedings of ACM SIGCOMM ASIA Workshop*, apr 2005.



- [82] J. Rosenberg, R. Mahy, and C. Huitema, “Traversal using relays around NAT (TURN): Relay extensions to session traversal utilities for NAT (STUN),” RFC-to-be 5766, Internet Engineering Task Force, feb 2010.
- [83] J. Rosenberg, “Interactive connectivity establishment (ICE): A protocol for network address translator (NAT) traversal for offer/answer protocols,” RFC-to-be 5245, feb 2010.
- [84] S. Perreault and J. Rosenberg, “TCP candidates with interactive connectivity establishment (ICE),” internet draft, Internet Engineering Task Force, oct 2008.
- [85] S. Guha, Y. Takeda, and P. Francis, “NUTSS: A SIP-based approach to UDP and TCP network connectivity,” in *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, p. 48, ACM, 2004.
- [86] J. Eppinger, “TCP connections for P2P apps: A software approach to solving the NAT problem,” tech. rep., Carnegie Mellon University, Jan. 2005.
- [87] Y. Wei, D. Yamada, S. Yoshida, and S. Goto, “A new method for symmetric NAT traversal in UDP and TCP,” *Asia Pacific Advanced Network (APAN)*, Aug. 2008.
- [88] Y. Chen and W. Jia, “Challenge and solutions of NAT traversal for ubiquitous and pervasive applications on the Internet,” *The Journal of Systems & Software*, vol. 82, no. 10, pp. 1620–1626, 2009.

- [89] J. Schmidt, “The hole trick, how Skype & Co. get round firewalls,” *heise Security UK*, Dec 2006.
- [90] S. Baset and H. Schulzrinne, “An analysis of the Skype peer-to-peer internet telephony protocol,” Sept. 2004.
- [91] B. Ford, P. Srisuresh, and D. Kegel, “Peer-to-peer communication across network address translators,” in *USENIX Annual Technical Conference*, (Anaheim, CA), April 2005.
- [92] A. Jantunen, S. Peltotalo, and J. Peltotalo, “Peer-to-Peer Analysis: State-of-the-art,” tech. rep., Tampere University of Technology, Feb. 2006.

# Vita

- Abdulaziz Muhammad Ali Al-Baiz
- Nationality: Saudi
- Born in Dammam, Saudi Arabia in 1984
- Received Bachelor of Science (B.S.) degree in Computer Engineering from King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia in June 2007.
- Received Master of Science (M.S.) degree in Computer Engineering from King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia in January 2011.
- Currently working as a system analyst at Saudi Aramco, EXPEC Advanced Research Center.
- Permanent Address: P.O. Box 3741, Dammam 31481, Saudi Arabia
- Phone number: (+966) 0505863918
- Email address: [aziz\\_baiz@hotmail.com](mailto:aziz_baiz@hotmail.com)