



GRADUATE SCHOOL
EAST TENNESSEE STATE UNIVERSITY

East Tennessee State University
Digital Commons @ East
Tennessee State University

Electronic Theses and Dissertations

Student Works

8-2003

Application Adaptive Bandwidth Management Using Real-Time Network Monitoring.

Amit Grover
East Tennessee State University

Follow this and additional works at: <https://dc.etsu.edu/etd>

 Part of the [Computer Sciences Commons](#)

Recommended Citation

Grover, Amit, "Application Adaptive Bandwidth Management Using Real-Time Network Monitoring." (2003). *Electronic Theses and Dissertations*. Paper 802. <https://dc.etsu.edu/etd/802>

This Thesis - unrestricted is brought to you for free and open access by the Student Works at Digital Commons @ East Tennessee State University. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of Digital Commons @ East Tennessee State University. For more information, please contact digilib@etsu.edu.

Application Adaptive Bandwidth Management Using
Real-Time Network Monitoring

A thesis
presented to
the faculty of the Department of Computer and Information Sciences
East Tennessee State University

In partial fulfillment
of the requirements for the degree
Master of Science in Computer Science

by
Amit Grover
August 2003

Dr Neil Thomas, Chair
Dr Phillip E. Pfeiffer IV
Dr Qing Yuan

Keywords: Bandwidth Management, Network, Security, Ports, Intrusion

ABSTRACT

Application Adaptive Bandwidth Management Using Real-Time Network Monitoring

by

Amit Grover

Application adaptive bandwidth management is a strategy for ensuring secure and reliable network operation in the presence of undesirable applications competing for a network's crucial bandwidth, covert channels of communication via non-standard traffic on well-known ports, and coordinated Denial of Service attacks. The study undertaken here explored the classification, analysis and management of the network traffic on the basis of ports and protocols used, type of applications, traffic direction and flow rates on the East Tennessee State University's campus-wide network. Bandwidth measurements over a nine-month period indicated bandwidth abuse of less than 0.0001% of total network bandwidth. The conclusion suggests the use of the defense-in-depth approach in conjunction with the KHYATI (Knowledge, Host hardening, Yauld monitoring, Analysis, Tools and Implementation) paradigm to ensure effective information assurance.

DEDICATION

To my parents Shri Suresh K Grover and Smt Asha Grover
and my brother Sumit Grover to whom I am
indebted for their constant encouragement and unwavering
support that helped me sail through trying times.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my advisory committee members—Dr Neil Thomas (chair for the final part), Dr Phillip E Pfeiffer IV (chair for the initial part), and Dr Qing Yuan—without whose unwavering support and guidance, this manuscript would not have been possible. Dr Pfeiffer’s academic brilliance, impeccable computing experience, clarity of vision, and meticulousness was responsible for shaping this thesis in its crucial foundation stages. Dr Thomas’s constant encouragement, thorough knowledge and invaluable computing expertise were responsible for guiding the remaining part of the thesis. I also wish to express my sincere thanks to the staff of OIT—Bob Cook, Steven Webb, Timothy White, JD Reed and Jim Ashe—for their help in dealing with various infrastructure issues.

CONTENTS

	Page
ABSTRACT	2
DEDICATION	3
ACKNOWLEDGEMENTS	4
LIST OF TABLES	11
LIST OF FIGURES.....	13
Chapter	
1 INTRODUCTION.....	15
1.0 Information Assurance.....	15
1.1 Synopsis of the Approach Taken.....	19
1.2 Firewalling.....	19
1.3 Packet-Shaping.....	19
1.4 Hybrid Strategy.....	20
1.5 Key Results.....	20
1.6 Structure for the Balance of the Thesis.....	21
2 PORT ABUSE AND COUNTERMEASURES.....	22
2.1 Characteristics of the Transport Layer	22
2.1.1 Mechanisms for Layer 4 Traffic Flow.....	22
2.1.1.1 Notion of Ports.....	23
2.1.1.2 Standard Uses of Ports.....	24
2.1.1.2.1 “Trojan” Ports.....	25
2.1.1.3 Covert Channels of Communication.....	25
2.1.2 Typical Characteristics of Layer 4 Traffic Flow.....	26

2.2	Abusing Ports to Compromise Security.....	27
2.2.1	Port Scanning.....	27
2.2.1.1	TCP Connect Scan.....	28
2.2.1.2	TCP SYN (Half Open) Scan.....	29
2.2.1.3	TCP FIN (Stealth) Scan.....	29
2.2.1.4	TCP Ftp Proxy (Bounce Attack) Scan.....	29
2.2.1.5	TCP Xmas Tree Scan.....	30
2.2.1.6	TCP Null Scan.....	30
2.2.1.7	TCP ACK Scan.....	30
2.2.1.8	TCP Windows Scan.....	31
2.2.1.9	TCP RPC Scan.....	31
2.2.1.10	SYN / FIN Scan Using IP Fragments (Bypass Packet Filters).....	31
2.2.1.11	UDP Recvfrom() And Write() Scan.....	32
2.2.1.12	UDP Raw ICMP Port Unreachable Scan.....	32
2.2.1.13	ICMP Echo Scan (Ping – Sweep)	32
2.2.1.14	Reverse-ident Scan.....	33
2.2.1.15	Idle Scan.....	33
2.2.2	Stack Fingerprinting.....	33
2.2.2.1	Active Stack Fingerprinting	34
2.2.2.1.1	FIN Probe.....	34
2.2.2.1.2	Bogus Flag Probe.....	34
2.2.2.1.3	Initial Sequence Number (ISN) Sampling...	35

2.2.2.1.4	“Don’t Fragment Bit” Monitoring.....	35
2.2.2.1.5	TCP Initial Window Size.....	35
2.2.2.1.6	ACK Value.....	36
2.2.2.1.7	ICMP Error Message Quenching.....	36
2.2.2.1.8	ICMP Message Quoting.....	36
2.2.2.1.9	ICMP Error Message-Echoing Integrity..	37
2.2.2.1.10	Type of Service (TOS)	37
2.2.2.1.11	Fragmentation Handling.....	37
2.2.2.1.12	TCP Options.....	38
2.2.2.2	Passive Stack Fingerprinting	38
2.3	Techniques for Detecting Invalid Use of Ports.....	38
2.3.1	Use Analysis.....	38
2.3.2	Bandwidth Analysis.....	39
2.3.3	Content Analysis.....	40
2.3.3.1	Real Time Intrusion Detection.....	42
2.3.3.1.1	Overload, Crash and Subterfuge Attacks.....	43
2.3.3.2	Traffic Normalization.....	44
2.3.3.3	Intrusion Detection Wrappers.....	45
2.3.3.4	Intrusion Detection Based On Expert Systems.....	45
2.3.3.5	Intrusion Prevention.....	46
2.3.4	Timing Analysis.....	46
2.4	Techniques for Blocking Invalid Use of “Mainstream” Ports.....	47
2.4.1	Shutting Down Unneeded Services.....	47

2.4.2	Port Re-Mapping.....	47
2.4.3	Static Packet Filtering.....	48
2.4.4	Dynamic Packet Filtering.....	48
2.4.5	Stateful Filtering.....	48
2.4.6	Proxy Filtering.....	49
2.4.7	Plug Gateways.....	49
2.4.8	Port Address Translation.....	49
2.4.9	Traffic Management.....	49
3	BANDWIDTH MANAGEMENT.....	51
3.1	Goals	51
3.2	Preliminaries.....	51
3.2.1	Background study.....	52
3.2.2	Data Collection Tools.....	52
3.2.2.1	PacketShaper.....	53
3.2.2.2	Active Ports	53
3.2.2.3	PacketPup	54
3.2.2.4	Network Probe 0.4.....	54
3.2.2.5	Ethereal	55
3.2.2.6	Action Request System	56
3.2.2.7	McAfee's GroupShield Exchange 5.0 Enterprise Suite...	56
3.2.3	Monitoring Station.....	56
3.3	Bandwidth Management.....	57
3.3.1	Classification of the Network Traffic.....	57

3.3.2	Analysis and Interpretation.....	61
3.3.3	Enforcement of Management Policies.....	63
3.3.3.1	Feedback-Oriented Fine-Tuning.....	65
3.3.4	Data Integrity Verification.....	73
3.3.5	Non-Standard Traffic	73
3.3.5.1	Step I: Initiate Network Probe.....	75
3.3.5.2	Step II: Isolate Communication for Interesting Ports.....	78
3.3.5.3	Step III : Identification of Potential Secondary Target Ports.....	78
3.3.5.4	Step IV : Isolating Conversations from ‘Suspect’ Hosts.....	79
3.3.5.5	Step V: Target Host Investigation.....	80
3.3.5.6	Step VI : Physical Identification of Target Machines....	84
4	RESULTS	88
4.1	Results	88
4.2	Bandwidth Traffic Tree.....	88
4.3	Top Ten Protocols.....	89
4.4	Breakdown of HTTP Traffic.....	90
4.5	Bandwidth Utilization.....	91
4.6	Rogue Traffic on Legitimate Ports.....	93
4.7	Bandwidth Abuse.....	94
4.7.1	Bandwidth Abuse Analysis.....	96
5	CONCLUSION.....	97

5.1	Defense–In–Depth	97
5.2	The ‘KHYATI’ Paradigm.....	97
5.2.1	Knowledge.....	98
5.2.2	Host hardening.....	98
5.2.3	Yauld monitoring.....	99
5.2.4	Analysis.....	99
5.2.5	Tools.....	99
5.2.6	Implementation.....	100
BIBLIOGRAPHY		101
APPENDICES		
Appendix A: PROTOCOLS AND SERVICES SUPPORTED BY ETHEREAL		105
Appendix B: PROTOCOLS AND SERVICES CLASSIFIED BY PACKETSHAPER		107
Appendix C: SAMPLE AVERAGE AND PEAK FLOW RATES.....		112
Appendix D: NETWORK PERFORMANCE SUMMARY.....		115
Appendix E: ETSU NETWORK TRAFFIC TREE AS ON APRIL 01 2003.....		122
Appendix F: PROTOCOLS AND SERVICES RECOGNIZED BY PACKETPUP.....		131
Appendix G: SAMPLE DATA FROM PACKETPUP LOGFILE.....		133
Appendix H: TOP TEN CLASS MEMBERS.....		136
Appendix I: MALICIOUS DATA ON LEGITIMATE PORTS.....		142
VITA.....		145

LIST OF TABLES

	Page
Table 1.1: Top Ten Probed Ports.....	17
Table 2.1: ISN sampling pattern Vs Stack implementation.....	35
Table 2.2: Alterations to packet headers made by various operating systems.....	37
Table 3.1: Summary of PacketShaper settings.....	57
Table 3.2: Top sending IP hosts in class /Outbound/Low/FTP.....	61
Table 3.3: Top receiving IP hosts in class /Outbound/Low/FTP.....	62
Table 3.4: Top sending IP hosts in class /Outbound/Average/pcANYWHERE.....	62
Table 3.5: Top receiving IP hosts in class /Outbound/Average/pcANYWHERE.....	63
Table 3.6: Partition Summary for ICMP Traffic.....	65
Table 3.7: Network Traffic Flow Data.....	75
Table 3.8: Hosts using NNTP	78
Table 3.9: All Protocols used by source host (netstat.etsu.edu).....	79
Table 3.10: All Protocols used by destination host (mail.etsu.edu).....	79
Table 3.11: Conversations originating from netstat.etsu.edu.....	79
Table 3.12: Conversations for destination host (mail.etsu.edu) using NNTP and SMTP ..	80
Table 3.13: Network card conversation.....	86
Table 4.1: Inbound traffic—Top 10 Protocols.....	89
Table 4.2: Outbound traffic—Top 10 Protocols.....	89
Table 4.3: Breakdown of HTTP Traffic.....	90
Table 4.4: Inbound Bandwidth Utilization.....	91
Table 4.5: Outbound Bandwidth Utilization.....	92
Table 4.6: Malicious Data in Legitimate Traffic.....	94

Table 4.7: Inbound Total Traffic vs. Rogue Traffic.....	95
Table 4.8: Outbound Total Traffic vs. Rogue Traffic.....	95

LIST OF FIGURES

	Page
Figure 2.1: Screenshot of the Windows version of ‘Nmap’	28
Figure 2.2: Screen shot of ‘Active Ports’ showing all open ports on a system.....	39
Figure 2.3: Screen shot depicting application-based monitoring of traffic.....	40
Figure 2.4: Screen shot showing ‘packet sniffing’ in action with detailed information. about captured data from the ETSU network.....	41
Figure 2.5: Protocol Hierarchy Breakdown.....	42
Figure 3.1: Network Probe in Action.....	55
Figure 3.2: Position of the Monitoring Station.....	58
Figure 3.3: Typical Hierarchical Network Traffic Classification Tree.....	59
Figure 3.4: Class Hits, Policy Hits and Traffic Volume for Different Classes.....	66
Figure 3.5: Monthly Average Rate (from 18 Sep 2002 to 18 Oct 2002).....	67
Figure 3.6: Monthly Peak Rate (from 18 Sep 2002 to 18 Oct 2002).....	68
Figure 3.7: Inbound Utilization (from 17 Oct 2002 to 18 Oct 2002).....	68
Figure 3.8: Outbound Utilization (from 17 Oct 2002 to 18 Oct 2002).....	69
Figure 3.9: Inbound Network Efficiency (from 17 Oct 2002 to 18 Oct 2002).....	69
Figure 3.10: Outbound Network Efficiency (from 17 Oct 2002 to 18 Oct 2002).....	70
Figure 3.11: Inbound Top 10 Classes (from 17 Oct 2002 to 18 Oct 2002).....	70
Figure 3.12: Outbound Top 10 Classes (from 17 Oct 2002 to 18 Oct 2002).....	71
Figure 3.13: Transaction Delay.....	71
Figure 3.14: Transaction Delay Distribution.....	72
Figure 3.15: PacketPup in Action.....	74

Figure 3.16: Execution of the Happy99 Trojan.....	77
Figure 3.17: Positive Signature Match for the Happy99 Trojan.....	82
Figure 3.18: Positive Signature Match for the Happy99 Trojan—Expanded Tree.....	83
Figure 3.19: Reconstructed Email Message Containing the Happy99 Trojan Signature.	84
Figure 3.20: Remedy in Action.....	85
Figure 4.1: Breakdown of HTTP traffic: Percentage distribution.....	90
Figure 4.2: Inbound Bandwidth Distribution.....	91
Figure 4.3: Outbound Bandwidth Distribution.....	92
Figure 4.4: Rogue Traffic On Legitimate Ports.....	94
Figure 4.5: Bandwidth Abuse.....	95

CHAPTER 1

INTRODUCTION

“It’s unimportant whether we take out a computer center [controlling key weapon systems, electrical grids and telecommunications] with a bomb.... or a denial of service program.”

Department of Defense spokesman discussing the cyber-attack strategy in the run up to Operation Iraqi Freedom¹.

“...if we fail on [cyber] defense, the nation would be at risk....”

Maj. Gen. J David Bryan, Commander of the Joint Task Force-Computer Network Operations and vice director of the Defense Information Systems Agency².

1.0 Information Assurance

Operation Iraqi Freedom has etched in stone the role of information warfare, as this was the first major armed conflict that depended heavily on IT. Even in a war and sanctions-ravaged economy like Iraq with hardly any IT infrastructure, email messages to high ranking Iraqi military officials³ were used to soften the Iraqis’ attitude towards US military action by encouraging them to surrender and to help in toppling Saddam Hussein’s regime.

Information Warfare is broadly classified into two branches, viz., Information Denial and Information Assurance. *Information assurance*, the study of how to ensure the availability and security of critical network operations at all times, is emerging as a key concern in military and corporate organizations alike. Maj. Gen Bryan, the coordinator of the Terminator project² (a network security tool being jointly developed by the DOD, National Security Agency and

Lancope Inc.) emphasizes the need for Terminator-like tools to deal more quickly and effectively with exploitation of vulnerabilities like the recent SQL Slammer worm.

This thesis is concerned with an aspect of information assurance: the use of *application adaptive bandwidth management* to counter the threats posed by undesirable and non-standard traffic on well-known ports. A network's *bandwidth* is the maximum amount of data that that network can carry, measured in bits per second (bps). *Bandwidth management* is the practice of allocating a network's bandwidth, based on considerations like perceived priority and fair use. *Application adaptive bandwidth management* is a form of bandwidth management that uses an application's perceived importance to allocate bandwidth, based on overall network traffic. Strategies for application adaptive bandwidth management attempt to ensure that the most mission critical applications get bandwidth—and that malicious codes do not.

This thesis describes the study of application adaptive bandwidth management and unauthorized use of standard network channels for communication. Covert channels of communication that use well known ports for non-standard traffic are a major threat to the security of any network. According to the Dshield⁴ website, the top ten ports probed over a period of one month (from 15 May to 15 June 2003) as shown in table 1.1 indicates the main targets to be port 137 (netbios-ns) and port 80 (HTTP). Although port 80—one of the most widely used ports—has been assigned for World Wide Web HTTP services, the following⁵ Trojans also use the same port: 711trojan, AckCmd, BackEnd, BO2000Plug-Ins, Cafeini, CGIBackdoor, Executor, GodMessage4Creator, GodMessage, Hooker, IISworm, MTX, NCX, Noob, Ramen, ReverseWWWTunnel, RingZero, RTB666, Seeker, WANRemote, WebDownloader and WebServerCT.

Table 1.1: Top Ten Probed Ports

Service Name	Port Number	Explanation
netbios-ns	137	NETBIOS Name Service
www	80	World Wide Web HTTP
ms-sql-m	1434	Microsoft-SQL-Monitor
microsoft-ds	445	Win2k+ Server Message Block
netbios-ssn	139	NETBIOS Session Service
eDonkey2000	4662	eDonkey2000 Server Default Port
smtp	25	Simple Mail Transfer
---	41170	
ident	113	
---	0	

An open port is a potential security hole that can be used by hackers to access computers. Covert channels of communication render the target network susceptible to remote administration. The potential for damage depends solely on the maliciousness of the attacker's intent. The damage can range from a complete loss of critical data to involuntary use of the network for illegal transmission of copyrighted digital content. A compromised network may also be used to carry out a coordinated denial of service attack. Unauthorized activity on the network, even if it involves 'un-harmful probing', competes for available network bandwidth, an important institutional resource. With the ever-increasing number of users and applications, it becomes difficult to ensure adequate bandwidth and quality of service for mission critical applications. According to a Carnegie Mellon University review⁶, "*...traditional Ethernet-and-IP network elements (routers and switches) perform well in lightly-loaded situations, but problems arise as traffic increases. For example, shared-access Ethernet rapidly degrades in throughput after about 30% utilization. In this situation, the number of collisions and retransmissions grows quickly, reducing the network efficiency...*" An increase in the link utilization might force the routers to drop the packets after a certain limit. Since resources are

limited, the cost factor rules out an indefinite upgrading of the network bandwidth capacity as a viable solution.

The goal of this thesis was to identify and minimize the use of excessive bandwidth by undesirable applications and minimize port abuse and flow of malicious data on the ETSU campus network. In a typical university setting, students may use bandwidth for file swapping utilities aimed at downloading large multimedia files (pirated movies and MP3s) and warez. Other activities such as IRC applications and MUD games might be used as a gateway by Trojans that can compromise network security by allowing remote administration. These activities steal bandwidth from critical primary applications (administrational/educational), jeopardize the overall network security, and may even put the host institution into legal jeopardy. In October 2002, for example⁷, four college students in San Jose, California were sued by the recording industry for exploiting the academic resources of their campus networks for file-swapping services. The Recording Industry Association of America accuses them of illegally swapping about a million songs without the permission of the copyright holders and seeks a maximum penalty of \$150,000 per song from each student. According to the news report, *“The recording industry telegraphed its campus crackdown last October putting 2,300 university administrators on notice to curb student behavior—or face legal consequences”*. At ETSU itself there have been “several instances of abuses” of network resources as pointed out by President Paul E Stanton’s memorandum⁸ regarding “Utilization of Computer Resources at East Tennessee State University”. The memorandum specifically prohibits all employees from indulging in “deliberate and wasteful use of [computer] resources” for unauthorized processing of data / files “that are not associated with their assigned duties”.

1.1 Synopsis of the Approach Taken

In the initial stages of the study itself, the lack of a single-piece solution became apparent. To achieve the objectives, I used a combination of various tools such as firewalls, packet-shapers, protocol analyzers, port scanners, network traffic monitoring tools, anti-virus programs and service management tools. These tools are covered in detail in sections 2 and 3. The classic approach for securing a network against malicious code involves port based filtering and the quest for a solution led me to a firewall—the ‘seemingly-obvious’ choice for blocking undesirable data as well as restricting covert channels of communication.

1.2 Firewalling

This strategy involves identifying non-standard ports generally used by Trojans and other malicious code as ‘rogue ports’ and then blocking these ports using firewalls. While firewalls can be used effectively to block known rogue ports, they are ineffective against undesirable applications such as file swapping utilities or Trojans that can use http traffic to infect the machine and then switch to un-allocated ports that are not being blocked by firewalls. Secondly, firewalls by themselves cannot provide bandwidth management by ensuring availability of bandwidth for critical applications. This necessitates the use of a packet-shaper in conjunction with the firewall.

1.3 Packet-Shaping

Packet shapers manage network bandwidth usage to a finer degree than what is possible with only switches or routers. They help to prioritize network usage by identifying ‘critical’ as well as ‘less desirable’ components of the traffic stream and earmarking more bandwidth for critical applications while restricting the amount of bandwidth available for less desirable

applications. Packet shapers classify network traffic by evaluating traffic flow on the basis of application-specific ports, protocol family (including transport protocols like TCP and UDP), IP address, port, the IP precedence value and URL. A CNN news report dated 10 October 2002 and titled *‘Student’s file sharing overloads college networks’*⁹ indicates that about 740 educational institutes in America were using Packeteer’s PacketShaper tool to manage bandwidth.

1.4 Hybrid Strategy

While a firewall would block data on Trojan ports and a packet shaper would allow bandwidth allocation for mission critical applications, their use does not solve the problem of non-standard traffic on standard desirable ports. This led me to explore the use of port scanners for identifying open ports and protocol analyzers for content based monitoring of the network traffic stream. Anti-virus tools were used for identifying known malicious code based on signature matching in the network traffic. While these tools allowed detection of malicious code in the traffic, to be able to trace the actual transmission route from the source to the destination, I relied on network traffic monitoring and service management tools. Thus the non-availability of a “silver-bullet”¹⁰ solution led to the use of a hybrid strategy that didn’t depend solely on any one single tool.

1.5 Key results

Yauld network monitoring, together with aggressive vulnerability management, helped to minimize bandwidth abuse on the ETSU network. Actual bandwidth measurements spread over a nine-month period from August 2002 to April 2003 indicated the percentage bandwidth abuse to be a mere 0.000073393%. These measurements were based on the actual network traffic inside the firewall and do not account for the substantial amount of undesirable traffic that is

blocked by the firewall itself. The bandwidth measurements highlighted HTTP traffic as the most predominant protocol in the traffic stream—constituting as high as 76% of the total traffic. Of the incoming malicious code that passed through the firewall, the Klez virus code was the most predominant with 25,591 distinct occurrences in the period under consideration. Detailed bandwidth measurements for incoming as well as outgoing traffic—organized as mission critical, desirable, non-critical, and rogue—are tabulated in the section 4. The lessons learnt made me conclude that effective information assurance in an environment like ETSU’s requires the use of the defense-in-depth security in conjunction with KHYATI (Knowledge, Host hardening, Yauld monitoring, Analysis, Tools and Implementation) network management. The defense-in-depth strategy relies on multiple layers of defense thereby eliminating one single point-of-failure for the entire system. Complementing this approach with the KHYATI paradigm would result in achieving information assurance goals in an effective and efficient manner.

1.6 Structure for the balance of the thesis

The balance of this thesis is divided into four sections. Section 2 discusses the concept of ports, how they can be abused to compromise security as well as techniques for detecting and blocking the invalid use of ports. Section 3 concentrates on specifics such as the tools and the methodology used to achieve the goals. Section 4 covers the results of the thesis in detail. Section 5 concludes with a discussion of defense-in-depth, the ‘KHYATI’ paradigm, and lessons learned.

CHAPTER 2

PORT ABUSE AND COUNTERMEASURES

2.1 Characteristics of the Transport Layer

The ISO's Open Systems Interconnection (OSI) Reference Model divides communications architectures into seven distinct layers. OSI Layer 4, the 'Transport' layer, is tasked with providing efficient and reliable end-to-end communication. According to Andrew S Tanenbaum¹¹,

"... (The Transport layer) is the heart of the whole protocol hierarchy. Its task is to provide reliable, cost-effective data transport from the source machine to the destination machine, independent of the physical network or networks currently in use. Without the transport layer, the whole concept of layered protocols would make little sense."

Currently, the dominant communications architecture for wide-area communication is TCP/IP (Transmission Control Protocol / Internet Protocol). In TCP/IP's reference model, the Transport layer uses the third, Internet layer to provide a channel of communication between the source and destination endpoints. An endpoint consists of an IP address, a protocol identifier and a port number¹².

2.1.1 Mechanisms for Layer 4 Traffic Flow

Currently, the two dominant transport layer protocols are TCP/IP's Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP is a reliable, connection-oriented protocol. UDP is a connectionless protocol that does not incorporate transmission error checking.

TCP¹³ connections are implemented as point-to-point full-duplex byte streams. The data exchange unit, known as a segment, consists of a 20-byte header pre-pended to a variable-length segment.

TCP ensures reliable data transmission using Automatic Repeat Requests (ARQs). An ARQ is an automatic re-transmission of data following a failure to receive a positive acknowledgement (ACK) from the receiving host.

TCP flow control is managed with a 'Sliding Window'¹⁴ protocol. A typical TCP implementation supports variable window sizes and the selective-repeat form of sender-initiated ARQ. To ensure effective congestion-control, an effort is made to estimate the round-trip transmission delay as accurately as possible.

UDP¹⁵ allows transmission of encapsulated raw IP datagrams without the overhead of establishing and releasing explicit connections. Since UDP lacks the reliability of TCP, UDP is ideal for situations where the rate of delivery is more important than reliable transmission without any packet loss. The size of the UDP segment header is 8 bytes.

2.1.1.1 Notion of Ports

Computers connected to the Internet communicate with each other via endpoints known as *sockets*. A socket address is an identifying number derived from the combination of an IP address and a virtual port number. Clearly defined source and destination IP addresses as well as source and destination port addresses are the prerequisites for establishing communicating across a network. TCP and UDP support 65536 virtual or software ports, each of which is identified by a 16-bit number in the range of 0 through 65535. The 48-bit combination of a host's IPv4 address and port number is referred to as a Transport Service Access Points or TSAP. The

transport layer protocols—TCP (Transmission Control Protocol) as well as UDP (User Datagram Protocol)—use these ports to form a virtual channel for information exchange. Any active port running a network based service is known as an ‘open’ port. A malicious user can use open ports to identify a target system’s attributes, and then use this information as a starting point for compromising that host’s security.

2.1.1.2 Standard Uses of Ports

The Internet Assigned Numbers Authority (IANA)¹⁶ associates various applications and services with specific port numbers and classifies the entire range of available ports into three categories—Well Known Ports, Registered Ports, and Dynamic and/or Private Ports.

Well-known ports, ports numbered from 0 through 1023, serve as contact addresses for various pre-defined services. Usually root privileges are required by the applications that service well-known ports. Commonly used ports are 21 (FTP), 22(SSH), 23 (TELNET), 25 (SMTP), 43 (Who Is), 53(DNS), 80(HTTP), and 137-139 (NETBIOS). A detailed list specifying all well-known ports and their associated services is available at <http://www.iana.org/assignments/port-numbers>.

Registered ports, ports numbered from 1024 through 49151, serve as the recommended ports for various services but are not bound to any particular service. Depending on the availability, a host might open a random port in this range to communicate over a network. Unlike the well-known ports, registered ports do not require system/root privileges for access. A detailed list specifying the Registered Ports and the services associated with them is available at the IANA website at <http://www.iana.org/assignments/port-numbers>.

Dynamic ports, ports numbered from 49152 through 65535, are typically used by applications that are not registered with the IANA. This range is of extreme significance for a system administrator as open ports in this range often indicate the presence of Trojan applications on a network host.

2.1.1.2.1 “Trojan” Ports

Ports typically used by Trojan programs are known as ‘Trojan Ports’. Fixed port numbers like TCP port 12345 (NetBus) and UDP port 31337 (Back Orifice) were typically used by earlier Trojans. Newer Trojans such as SubSeven use a wide range of port numbers. Common Trojan horse port assignments include 1243 (Sub-7, SubSeven), 6670 (DeepThroat), 6711 (Sub-7, SubSeven), 6969 (GateCrasher), 12345 (NetBus), 23456(EvilFtp), 27374(Sub-7, SubSeven), 30100(NetSphere), 31789 (Hack'a'Tack), and 31337 (BackOrifice). Detailed lists of Trojan ports are available at www.doshelp.com/trojanports.htm, <http://www.simovits.com/trojans/trojans.html> and www.commodon.com/threat/threat-ports.htm.

2.1.1.3 Covert Channels of Communication

One of the most common methods to evade detection is to use covert channels of communication. One form of covert channel involves the use of non-standard ports to avoid detection. Most Intrusion Detection Systems (IDSs)¹⁷ typically monitor traffic on ports associated with standard protocols like DNS, IMAP, POP, SNMP, SYSLOG, TELNET, RLOGIN, RSH, FTP, or on ports associated with known backdoors / Trojans. Once a port is identified as being associated with malicious code, traffic through that port can be blocked using

a firewall. This led to the development of the newer generation of Trojans that dynamically choose a port from a pre-defined range.

A second kind of covert channel, aimed at subverting firewall-based filtering, uses standard ports for passing non-standard traffic. Firewalls that enforce a “block-all-but-necessary” approach to regulating traffic are the typical targets of standard port abuse. A recent (25 Jan 2003) case of standard port abuse involved a Denial of Service (DOS) attack that was variously known as the ‘SQL Slammer’ worm, ‘Sapphire’ and “SQL-Hell’. The worm in question used a vulnerability in Microsoft SQL Server to subvert targeted systems. The worm spreads via UDP port 1434, which is officially assigned for ‘Microsoft-SQL-Monitor’ services. The infected host starts transmitting 376 byte long UDP packets at a very high rate to random IP addresses on the Internet thereby generating overwhelming traffic. The attack caused widespread denial of service and adversely affected online services throughout the world.

2.1.2 Typical Characteristics of Layer 4 Traffic Flow

Efficient bandwidth management requires a thorough understanding of layer-4 traffic based on actual flows as opposed to simulated flow characteristics. In 1997, Kevin Thompson, Gregory Miller and Rick Wilder¹⁸ used traffic monitoring tools in MCI’s segment of the Internet backbone and the NSF sponsored vBNS to characterize typical layer 4 traffic “in terms of traffic volume, flow volume, flow duration, and traffic composition in terms of IP protocols, TCP and UDP applications, and packet sizes”. According to their observations,

- (a) TCP traffic accounted for 95% of bytes, 85-95% of packets and 75-85% of the flows. The remaining IP traffic was predominantly UDP while ICMP accounted for less than 1% of all packets.

- (b) Average packet size varied over time but followed a 24-hr pattern.
- (c) About 40% of all packets were 40 bytes long indicating TCP ACKs, FINs or RSTs.
- (d) Maximum application-based traffic was attributed to HTTP. Other TCP applications like FTP, NNTP etc. rarely exceeded 10% of the total traffic.
- (e) For UDP, the maximum traffic varied between DNS traffic and RealPlayer services depending on the time of the day.

2.2 Abusing Ports to Compromise Security

Open ports serve as potential access points for compromising a target host's security. Once a host's reachability has been verified using a utility like ping, a typical attack attempts to identify the target host's software, as a first step in exploiting known security vulnerabilities¹⁹. This task of software identification is achieved by mapping open ports to associated services. Two common techniques for getting this information include port scanning and fingerprinting.

2.2.1 Port Scanning

Port scanning is the discovery of open (listening) ports on a target host in order to ascertain the operating system and other services and applications running on the host. This typically involves scanning for TCP as well as UDP ports. One of the best-known port scanners, Fyodor's *nmap*²⁰, was used for collecting data for this thesis. A sample screen shot of nmap as shown below indicates the capability of this tool.

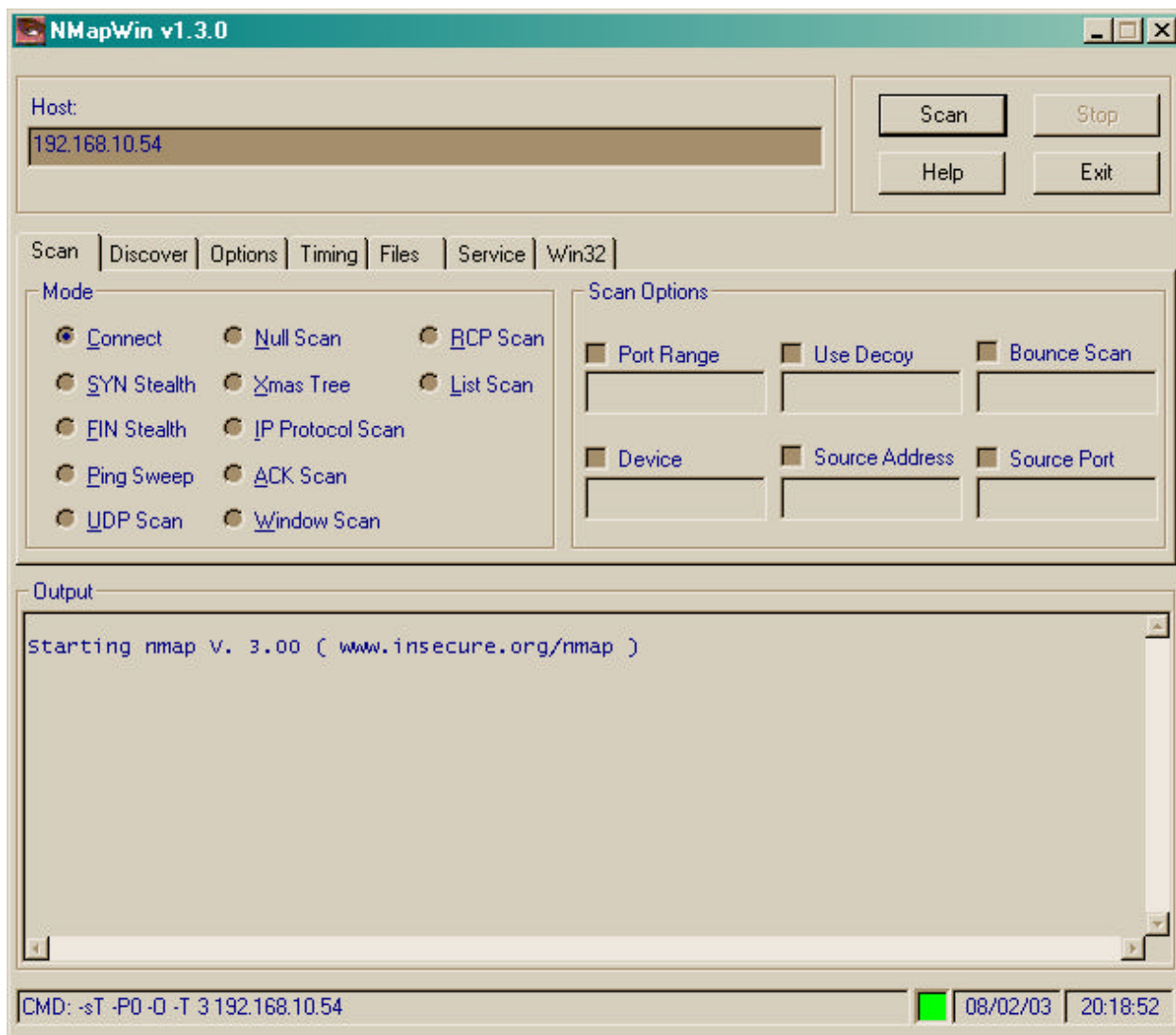


Figure 2.1: Screenshot of the Windows version of 'Nmap'

Various types of common port scanning techniques are described below.

2.2.1.1 TCP Connect Scan

This is the most fundamental type of scanning and involves the establishment of a TCP connection via its three-way handshake. The advantages of TCP connection scanning includes speed, in that scans can be done in parallel, and convenience, in that no special system privileges

are required for scanning. However this type of scanning is easily detectable and any perimeter-monitoring device on the target host would eventually block access to the originator of these scans.

2.2.1.2 TCP SYN (Half Open) Scan

SYN scans establish an incomplete connection with the target host. A SYN scan begins by sending a SYN packet to a target host. This packet evokes either a SYN/ACK response (if a particular port is open) or a RST response (if that port is closed). If the scanning host receives a SYN/ACK, it completes the scan in a non-standard way: i.e., with an RST to the target host, which terminates the connection. SYN scanning is therefore also known as "half-open" scanning. This technique is harder to detect than TCP connect scanning, but requires root privilege.

2.2.1.3 TCP FIN (Stealth) Scan

TCP FIN scanning (also known as 'stealth' scanning) was developed to evade detection by firewalls and static packet filters that detect TCP SYN scanning. TCP specifications require open ports to ignore FIN packets and closed ports to reply to FIN packets with the proper RST. However, Microsoft's TCP stack replies with a RST irrespective of the status of the port. This fact enables FIN scanning to differentiate between UNIX and Microsoft implementations.

2.2.1.4 TCP Ftp Proxy (Bounce Attack) Scan

Ftp proxy scanning exploits a security flaw in the FTP protocol (RFC 959). Proxy ftp connections allow an intruder to connect to an ftp server behind a firewall and execute read-write operations. Any port scanning that is done using a proxy ftp server could then bypass the

firewall-filtering rules, as the requests now seem to be generated from inside the network. This ability to carry out untraceable port scans makes ‘bounce attack’ scanning very attractive to potential intruders. The scanner typically connects to an anonymous ftp server and then scans the TCP ports from that proxy ftp server.

2.2.1.5 TCP Xmas Tree Scan

Xmas Tree scans send TCP packets with the FIN, URG and PSH flags set. A response of RST from the target host indicates that particular port as closed. No response indicates that the port is open and listening. The response can be used as one of the factors in determining the target host’s operating system on the basis of established RFC 793—compliance information for different stack implementations.

2.2.1.6 TCP Null Scan

Null scans send TCP packets with none of the flags set identify closed ports on the basis of a RST response from the host. RFC 793 specifies that open ports should ignore TCP NULL packets. However, certain TCP/IP stack implementations such as IRIX, HP-UX, Cisco IOS, MVS and Microsoft Windows are not fully compliant. This lack of full compliance results in different response to a TCP NULL scan by different operating systems and can be used for fingerprinting.

2.2.1.7 TCP ACK Scan

TCP ACK scans send TCP ACK packets to a target host. ACK scanning will elicit responses from hosts that might have been configured to ignore ICMP pinging. Scans of port 80

(HTTP) can identify active web sites that block ICMP echo requests (pings): an open port will elicit a RST response. This type of scanning is slightly different in that it seeks to determine whether the firewall protecting the target host is based on simple rules or employs advanced packet filtering techniques.

2.2.1.8 TCP Windows Scan

This technique seeks to identify open ports on a target system based on a system's window size characteristics. This exploits an anomaly in the AIX and FreeBSD systems in reporting the TCP window size. Depending on the operating system of the target host, the TCP windows scan might yield information about the status of certain filtered as well as non-filtered ports.

2.2.1.9 TCP RPC Scan

Sun Microsystems' RPC (Remote Procedure Call) mechanism simplifies distributed client-server computing by allowing network communications to be framed as subroutine calls. The ports designated for RPC start at port number 32678. TCP RPC scans detect and identify ports being used by Remote Procedure Call services. This scanning technique also provides information on the version number and programs associated with the RPC services running on a target host.

2.2.1.10 SYN / FIN Scan Using IP Fragments (Bypass Packet Filters)

SYN/FIN scans fragment probe packets before sending them, in an attempt to avoid detection by firewalls and packet filters. The TCP header is fragmented over a number of packets

to bypass the filtering mechanism. While this approach works with many firewalls, it will not work against systems that queue all IP fragments and reassemble the incoming packets thus identifying the active status of the SYN / FIN flags.

2.2.1.11 UDP Recvfrom() And Write() Scan

Users without root access to a scanning platform can `recvfrom()` and `write()` scans to identify open ports on some target hosts. The Linux UDP stack responds to an unsolicited `readfrom()` with error number 13 (EAGAIN- "try again") and error number 111 (ECONNREFUSED- "Connection refused"), according to whether the targeted port is open or closed.

2.2.1.12 UDP Raw ICMP Port Unreachable Scan

Raw ICMP port unreachable scans are similar to TCP bounce attack scanning, but target the UDP protocol. The scan attempts to distinguish between closed and open ports by treating an `ICMP_PORT_UNREACH` error message as a response from a closed port, and no response as an indicator of an open port. However, this technique is not very reliable, as some UDP stack implementations don't respond to either. Root privilege is required for initiating UDP ICMP port unreachable scanning.

2.2.1.13 ICMP Echo Scan (Ping – Sweep)

Strictly speaking, ICMP Echo Scanning is not a port scanning technique, in that ICMP does not support ports. This technique seeks to determine the active hosts by using the ICMP echo request (ping) command. An ICMP echo reply message (type value = 0; code value = 0) in

response to an ICMP echo request message (type value = 8; code value = 0) indicates that the target host is alive and that the ICMP messages are not being filtered. A number of port scanners now support parallel ICMP scanning, making the process of scanning an entire network or a range of hosts time-efficient. Nmap supports non-blocking I/O and parallel scanning in all TCP and UDP modes.

2.2.1.14 Reverse-ident Scan

This technique uses TCP's ident protocol (RFC 1413) to determine the owner of an open port's server process. Unlike the fragmentation scans, the TCP reverse ident scan requires a complete TCP connection with the target port. Potential attackers can use this feature to determine if a port is being serviced by a task with root privileges—and, if so, the task's associated account. The user name thus determined can be used for getting more information about the network.

2.2.1.15 Idle Scan

Idle scanning is a highly sophisticated port scanning technique invented by Antirez²². In idle scanning, the attacker sends probes from a 'dumb' host; thereby giving a fictitious IP address to any Intrusion Detection System that detects the scanning. Because of its high stealth capability, it is also known as "blind port scanning". Idle scanning can additionally be used to establish IP-based share relationships between trusted hosts on a network.

2.2.2 Stack Fingerprinting

According to Stuart McClure, Joel Scambray and George Kurtz²³, “*stack fingerprinting is an extremely powerful technology for ascertaining each host’s operating system with a high degree of probability*”. Stack fingerprinting uses differences between vendor implementations of TCP/IP stacks to identify target host software. Stack fingerprinting can be active or passive.

2.2.2.1 Active Stack Fingerprinting

In active stack fingerprinting, an attacker probes a target host’s open ports, then compares the responses to a database of known ‘signature-behavior mappings’ to profile the target host. According to McClure et al, active stack fingerprinting can be achieved by using the various types of probes listed below:

2.2.2.1.1 FIN Probe

FIN probes were discussed above, in Section 2.2.1.3.

2.2.2.1.2 Bogus Flag Probe

Bogus flag probes monitor the response to a SYN packet with an undefined TCP flag set (bit 7 or 8) in the header. Some operating systems reset the connection on receiving a TCP packet with bogus flags set; others, like older Linux versions (ver. 2.0.35 and earlier) responded by mirroring the bogus flags in their response packet headers. Of late, however, the 8th bit is being used for the “ECN field” for TCP congestion control. The port scanner “Queso” was one of the first scanners to exploit this technique for OS-determination.

2.2.2.1.3 Initial Sequence Number (ISN) Sampling

ISN sampling looks for patterns in the initial sequence numbers (ISNs) from target hosts. Operating systems may be identified on the basis of the sampling pattern of the response. Some known patterns along with their associated stack implementations are placed in table 2.1.

Table 2.1: ISN sampling pattern Vs Stack implementation

#	Sampling Pattern	Stack Implementation
1	Traditional 64K	SCO Unix (and most earlier versions of Unix)
2	Random incremental	FreeBSD, Digital UNIX, Cray, Solaris, IRIX, and newer versions of Unix
3	True "random"	Linux 2.0.*, OpenVMS, newer AIX
4	Time dependent	Microsoft Windows
5	Constant	Apple LaserWriter printers and 3Com hubs

2.2.2.1.4 “Don’t Fragment Bit” Monitoring

Certain operating systems such as Solaris set the “don’t fragment bit” on the IP packets that they transmit. Different stack implementations handle this bit differently. The setting of this bit is monitored and compared with a known database to aid estimation of the target operating system.

2.2.2.1.5 TCP Initial Window Size

Certain operating system stacks implement unique initial window sizes and this can be used as a signature attribute. While Microsoft Windows 2000, FreeBSD and OpenBSD use 0x402E; AIX uses 0x3F25. The initial TCP window size of the returned packets can thus help in OS determination of the target host.

2.2.2.1.6 ACK Value

ACK Value probing examines ACK field sequence values, yet another signature attribute. On sending a TCP packet with the FIN, PSH, and URG flags set to a closed port, Microsoft Windows will send an ACK with the initial sequence number incremented by one whereas most Unix-based operating systems will send an ACK with the same ISN set as the probe packet. On sending a TCP packet with the SYN, FIN, PSH, and URG flags set to an open port, Microsoft Windows will respond with an ACK packet with a random value for the ISN.

2.2.2.1.7 ICMP Error Message Quenching

Some operating systems such as Linux limit the generation of error messages (e.g. destination unreachable message) in accordance with the recommendations of RFC 1812. The ICMP error message quenching scan involves sending UDP packets on random ports and inferring the identity of the target's host stack from the number of unreachable messages in a given time period.

2.2.2.1.8 ICMP Message Quoting

In accordance with the recommendations of the various RFCs, ICMP error messages contain part information regarding the error-causing packet. While most stack implementations send 8 additional bytes along with the IP header, Solaris and Linux typically return more information. ICMP error message quoting infers operating system identity from the diagnostic returned in response to an ICMP error as an estimation attribute. Using known responses to various errors, it might be possible to detect hosts running Linux as well as Solaris systems even if all their ports are closed.

2.2.2.1.9 ICMP Error Message-Echoing Integrity

Echo integrity probing checks for changes to scanner-generated IP headers that are returned from the target host, in ICMP error messages. Many stack implementations inadvertently modify the packet header while processing them. Comparing the alterations made with an existing database of ‘alteration-signatures’ can pinpoint the target operating system with accuracy. Some commonly known alterations are tabulated in Table 2.2.

Table 2.2: Alterations to packet headers made by various operating systems

#	Characteristics of returned header	Operating system
1	Alterations to the IP ID	BSDI, FreeBSD, OpenBSD, ULTRIX, and VAXen
2	Checksum errors	AIX and FreeBSD
3	'Total length' field is 20 bytes longer than normal	AIX and BSDI

2.2.2.1.10 Type of Service (TOS)

The value in the TOS returned for “ICMP port unreachable” messages might vary for different stack implementations. While most implementations return ‘0’ for ICMP port unreachable messages, Linux returns a value of ‘0xC0’.

2.2.2.1.11 Fragmentation Handling

Different operating systems handle overlapping fragments of IP packets differently. The reassembled packets can give information about the reassembly process used. Fragmentation handling involves examining the method of reassembling of the probe packet-fragments to estimate the stack implementation.

2.2.2.1.12 TCP Options

TCP options probing checks how target systems manage requests for “custom” TCP options, like window scale factor, timestamps, maximum segment size, no operation, and end of operation. (cf. RFC 793 and RFC 1323). These options, which are not mandatory as per the RFC, are only implemented by some TCP/IP stacks.

2.2.2.2 Passive Stack Fingerprinting

In passive fingerprinting, the attacker maps ports without specifically probing the target host. The target machine’s TCP/IP stack is reconnoitered by observing TCP/IP session attributes like TTL, ‘Don’t fragment’ bits, and window size. Passive fingerprinting, though not as accurate as active fingerprinting, is more difficult to detect.

2.3 Techniques for Detecting Invalid Use of Ports

Port abuse is detected by identifying malicious actions and abnormal behavior that might compromise the integrity, confidentiality, or availability of information resources. The various techniques can broadly be classified as use analysis, bandwidth analysis, content analysis and timing analysis.

2.3.1 Use Analysis

Use analysis checks for port abuse by monitoring the numbers of the actual ports in use. Services known to be associated with distinctive port numbers can be identified using a pre-established database to map services to port numbers. Commonly used tools for this technique

include “netstat” as well as “Active Ports”. A sample screen shot of Active Ports listing all open ports on a host is given below:

Process	PID	Local IP	Local Port	Remote IP	Remote Port	State	Protocol	Path
System	0	151.141.8.209	137			LISTEN	UDP	
System	8	151.141.8.209	138			LISTEN	UDP	
System	8	0.0.0.0	445			LISTEN	UDP	
System	8	151.141.8.209	1727	151.141.8.100	445	ESTABLISHED	TCP	
System	8	151.141.8.209	1723	151.141.8.46	445	ESTABLISHED	TCP	
System	8	151.141.8.209	1693	151.141.8.22	445	ESTABLISHED	TCP	
System	8	0.0.0.0	4529			LISTEN	TCP	
System	8	0.0.0.0	1129			LISTEN	TCP	
System	8	0.0.0.0	1091			LISTEN	TCP	
System	8	151.141.8.209	1713	151.141.8.158	445	ESTABLISHED	TCP	
System	8	151.141.8.209	139			LISTEN	TCP	
System	8	151.141.8.209	1707	151.141.8.101	445	ESTABLISHED	TCP	
System	8	151.141.8.209	1615	151.141.8.100	445	ESTABLISHED	TCP	
System	8	0.0.0.0	445			LISTEN	TCP	
WINLOGON...	180	127.0.0.1	1043			LISTEN	UDP	C:\WINNT\system32\WINLOGON.EXE
services.exe	208	0.0.0.0	1048			LISTEN	UDP	C:\WINNT\system32\services.exe
lsass.exe	220	151.141.8.209	500			LISTEN	UDP	C:\WINNT\system32\lsass.exe
lsass.exe	220	127.0.0.1	1026			LISTEN	UDP	C:\WINNT\system32\lsass.exe
svchost.exe	368	0.0.0.0	135			LISTEN	UDP	C:\WINNT\system32\svchost.exe
svchost.exe	368	0.0.0.0	135			LISTEN	TCP	C:\WINNT\system32\svchost.exe
naimas32.exe	540	0.0.0.0	8084			LISTEN	TCP	C:\Program Files\naimas32.exe
MSTask.exe	644	0.0.0.0	1054			LISTEN	TCP	C:\WINNT\system32\MSTask.exe
Wuser32.exe	760	0.0.0.0	2702			LISTEN	TCP	C:\WINDOWS\MS\MSVCLID\MP\RemDir\Wuser32.exe
Wuser32.exe	760	0.0.0.0	2701			LISTEN	TCP	C:\WINDOWS\MS\MSVCLID\MP\RemDir\Wuser32.exe

Fig 2.2: Screen shot of ‘Active Ports’ showing all open ports on a system.

Rigorous logging and analysis of all remote-connection attempts is another way of detecting potential attacks.

2.3.2 Bandwidth Analysis

Bandwidth analysis is an important technique for identifying abnormal network usage. Packeteer Inc.’s ‘PacketShaper’ was used extensively to collect data for this thesis. PacketShaper allows application adaptive real time monitoring of network traffic and gives the ability to monitor/filter traffic on a port-to-port basis. A detailed account of the packet shaper is given in section 3. A sample screen shot depicting application-based monitoring using Packet Shaper is shown below:

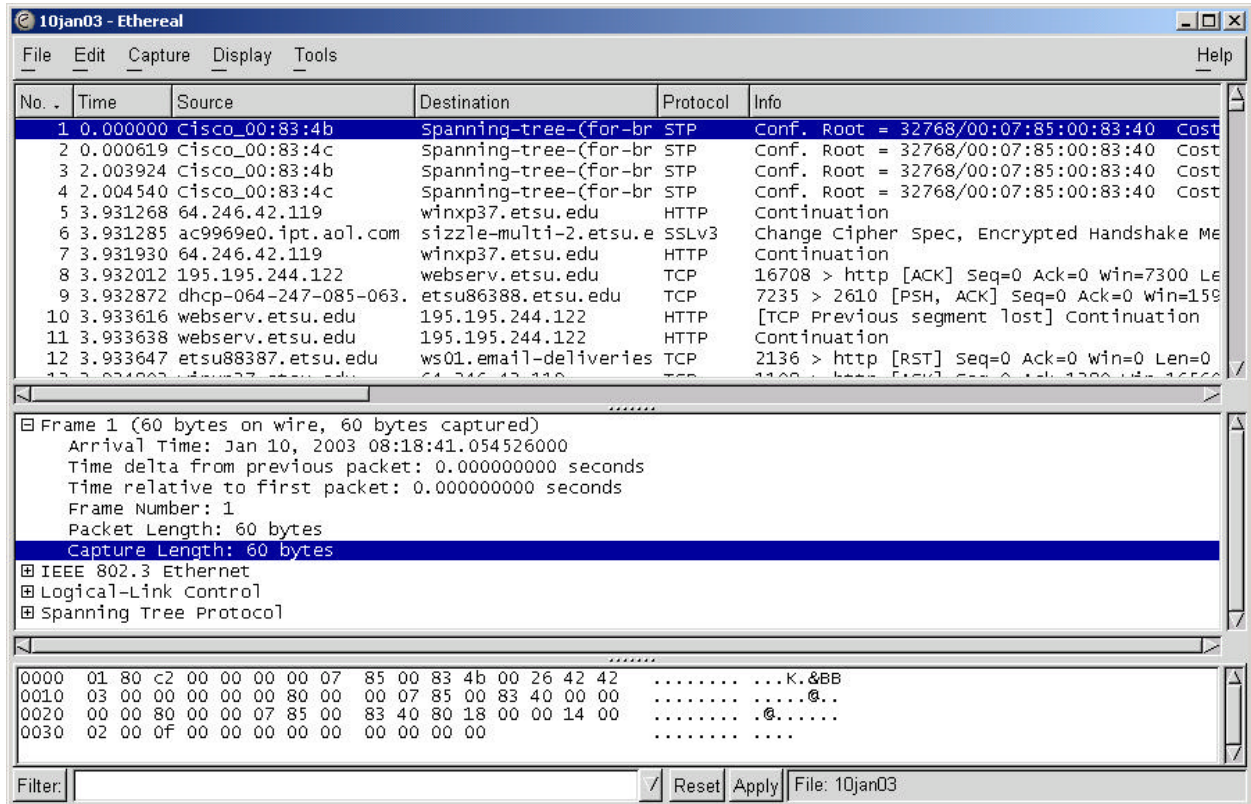


Fig 2.4: Screen shot showing ‘packet sniffing’ in action with detailed information about captured data from the ETSU network.

A detailed protocol hierarchy breakdown of the captured data is shown below in yet another screen shot from Ethereal.

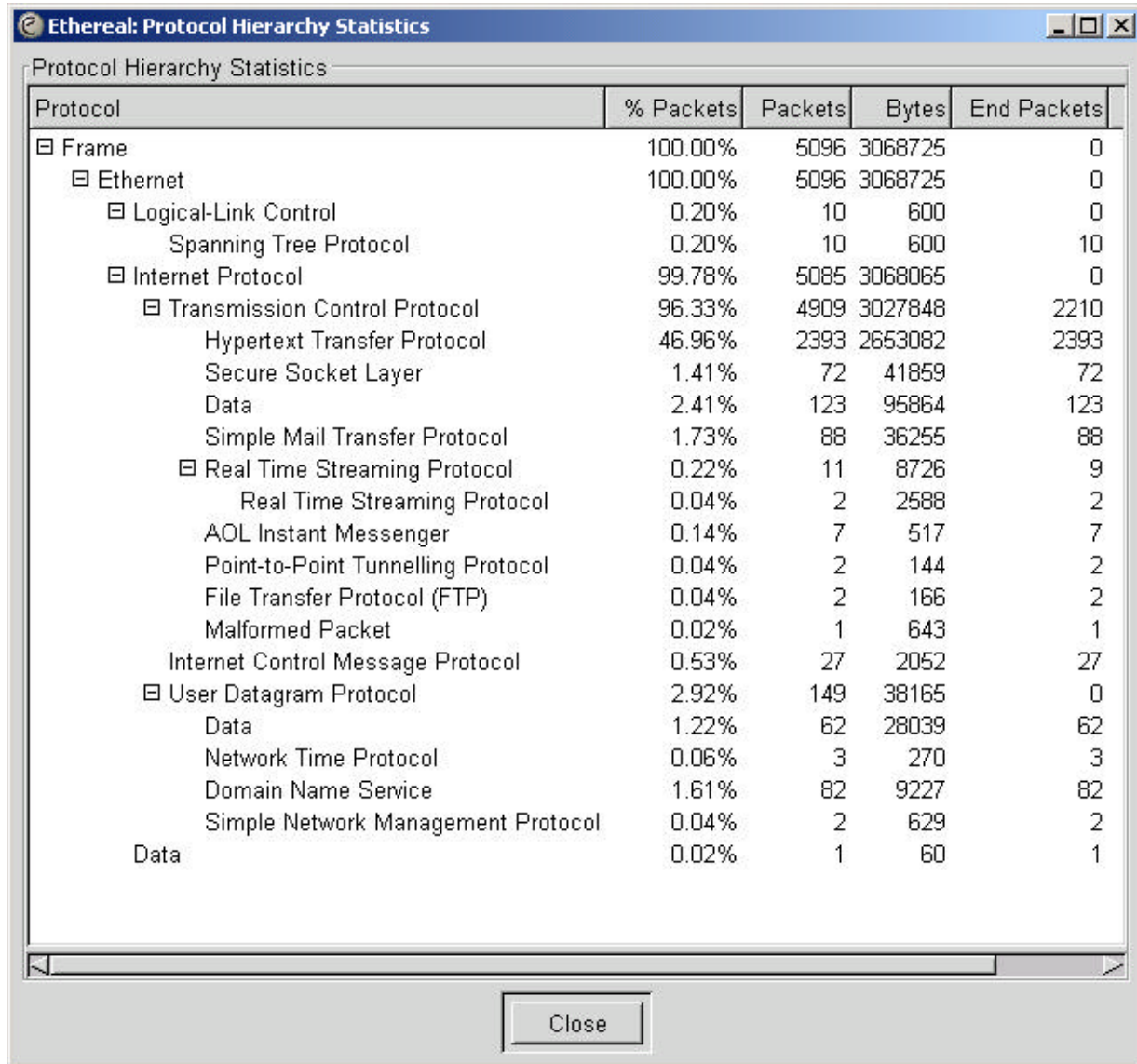


Fig 2.5: Protocol Hierarchy Breakdown

2.3.3.1 Real Time Intrusion Detection

Real time intrusion detection²⁶ systems such as Vern Paxson's 'Bro' can effectively defend against overload, crash as well as subterfuge attacks against the network host. Bro

consists of an *event engine* that converts a stream of filtered packets to high-level network events, and an *interpreter* for a specialized language for writing the security policy.

Bro has a layered structure. Its lowest layer, *libpcap*, is the packet capture directory used by the *tcpdump* protocol analyzer utility. This layer isolates Bro from the network link technology and enhances Bro's portability. The filtered packet stream from *libpcap* is passed to the event engine, which is the next layer. The event engine confirms the integrity of packet headers. All packets failing this check are discarded. Packets passing the integrity check are sent for further processing in separate streams for TCP and UDP packets. The processing involves invoking a handler to process the data payload of the packet. The processed event stream is then passed to the policy script interpreter, which generates a real-time notification of intrusions. Bro is specifically designed to handle high speed (FDDI-rate) large volume monitoring with an emphasis on extensibility and on avoiding packet filter drops.

2.3.3.1.1 Overload, Crash and Subterfuge Attacks

Depending on the events generated by the event engine's processing of a data packet, Bro executes event-handler commands until the queue is empty. Paxson claims that Bro has been designed to survive *overload*, *crash* and *subterfuge* attacks against the network monitor.

An *overload attack* first overwhelms the monitor with excessive data to ensure that it is unable to keep with the data stream. The actual network intrusion starts after the monitor has been overwhelmed. As long as the attacker doesn't have access to the policy scripts, Bro provides reasonable defense against overload attacks.

A *crash attack* incapacitates the monitor by exhausting its resources. The actual network intrusion takes place after the monitor has been rendered useless. Bro has been provided with specific capability to avoid crash attacks.

A *subterfuge attack* depends on the attacker's ability to mislead the network monitor regarding the nature of the traffic it analyses. The nature of these attacks makes it extremely difficult to detect or prevent subterfuge attacks. Successful subterfuge attacks rely on modifying the traffic pattern in a manner that renders the traffic stream open to different interpretations by the network monitor and the intended recipient. To protect against subterfuge attacks, Bro analyzes a system's rules for classifying network content for flawed assumptions. Paxson gives the example of texts embedded with a NUL to persuade the network monitor to ignore data thereafter and fragmenting the IP datagrams in a manner that might not be reassembled correctly by the monitoring device. Paxson claims that this equips Bro with a formidable intrusion detection capability while admitting that it still doesn't provide absolute security.

2.3.3.2 Traffic Normalization

Bro, which has its strengths, can unfortunately be bypassed by exploiting ambiguities in the network traffic stream. In "Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics", Handley²⁷, Paxson, and Kreibich describe the use of a network-forwarding element called a traffic normalizer to eliminate the traffic ambiguities by 'patching-up' or 'normalizing' the packet stream. Normalizers are similar to protocol scrubbers²⁸ but cover a wider range of conditions, and are optimized to defend coordinated attacks against the normalizers themselves.

2.3.3.3 Intrusion Detection Wrappers

In “Detecting and Countering System Intrusions Using Software Wrappers”, Ko²⁹, Fraser, Badger and Kilpatrick suggest the use of *ID wrappers*, in conjunction with intrusion detection techniques, to intercept problem events and take countermeasures if an event suggests the possibility of an intrusion. According to these authors, “*an ID wrapper is a software layer dynamically inserted into the kernel that can selectively intercept and analyze system calls performed by processes as well as respond to intrusive events*”.

ID wrappers facilitate program monitoring to detect unauthorized modifications that might be overlooked by traditional audit mechanisms. Wrapper countermeasures include denial, transformation or augmentation of the event. ID wrappers may also generate specialized events that would be intercepted by other intrusion detection wrappers in the system. The various ID wrappers are configured and managed by the ‘wrapper support subsystem’. Multiple ID wrappers can be used in a layered composition for more effective intrusion detection. The ‘*Common Intrusion Detection Framework*’ uses multiple ID wrappers to enhance performance.

2.3.3.4 Intrusion Detection Based on Expert Systems

Expert systems depend on extensive knowledge bases and sophisticated *IF-THEN-ELSE* rule sets. Depending on the status of critical events, they are designed to initiate actions in conformance to their driving rule sets. Ulf Lindqvist and Phillip A Porras³⁰ have described the use of PBEST (Production Based Expert System Toolset)—an expert system development toolset to develop a generic signature based IDS specifically for detecting SYN flooding and buffer overflows.

2.3.3.5 Intrusion Prevention

The ability to detect and isolate attacks before they compromise a system's core functionality is a prerequisite for building systems capable of surviving directed attacks. *Intrusion prevention*, as described by R. Sekar³¹ and P. Uppuluri, uses patterns of system calls to monitor and disallow calls that deviate from specifications. Since a potentially damaging system call can be modified before the damage is done, this technique gives the system time to prevent the damage.

Sekar and Uppuluri's intrusion prevention system consists of an offline and a runtime component. The offline system generates detection engines from specifications characterizing normal and abnormal program behavior as patterns of system call sequences, and the runtime system provides the execution environment for these engines. The authors claim that their algorithm, whose runtime is almost independent of the number of patterns, is suitable for any intrusion detection method involving pattern matching.

2.3.4 Timing Analysis

Yin Zhang³² and Vern Paxson, in "Detecting Backdoors", describe a generalized timing-based algorithm for detecting interactive network traffic using non-standard ports. The algorithm's goal is to detect the use of such ports to facilitate re-entry into systems that have already been breached. The authors use a timing-based algorithm instead of a content-based algorithm to render their work insensitive to encryption and decryption, and to reduce the algorithm's overhead. The algorithm accounts for packet size and directionality and the packet inter-arrival timing. The directionality helps in filtering the network traffic based on the premise that an interactive connection is always initiated by the client.

A high frequency of false alarms generated by this generalized algorithm prompted Zhang and Paxson to develop fifteen protocol-specific algorithms that detect backdoors on the basis of protocol-specific signatures. Depending upon the protocols or services used, the authors have classified the backdoors into various types—viz., SSH, Rlogin, Telnet, FTP, Root prompt, Napster and Gnutella—and developed specific algorithms to deal with each type.

2.4 Techniques for Blocking Invalid Use of “Mainstream” Ports

Authors like John E Canavan³³, Rolf Oppliger³⁴, Chris Benton and Cameron Hunt³⁵ identify a variety of techniques for blocking undesirable traffic. Nine of the more important of these techniques are discussed below.

2.4.1 Shutting Down Unneeded Services

Shutting down unneeded services is the simplest and most basic technique for preventing port abuse. Since open ports are potential access points on the network, care should be taken to disable all unnecessary services thereby closing all unnecessary ports on the host. Though this technique is the simplest to implement, it fails to provide a reasonable level of security as malicious programs target commonly used ports for compromising a target host’s security.

2.4.2 Port Re-Mapping

Port re-mapping involves running important services on non-standard ports thereby increasing the degree of difficulty involved in compromising such a system. This technique is effective only against ‘script-kiddies’ as experienced and dedicated hackers are generally able to

figure out if certain services are running on non-standard ports. If port re-mapping is not done properly, it might actually increase the system's vulnerability as new ports are opened.

2.4.3 Static Packet Filtering

Static packet filtering involves filtering incoming as well as outgoing data packets on the basis of the header information. The packets are denied or allowed access depending on the access control policies defined for the system. Static filtering controls traffic flow by analyzing information such as the source and destination IP address or subnet, the source and destination port addresses and the TCP flag field. This technique is difficult to implement for controlling UDP traffic, as the UDP header does not use flags to indicate a session's state.

2.4.4 Dynamic Packet Filtering

Dynamic packet filtering enhances static packet filtering by controlling traffic based on packet attributes as well as connection-state monitoring. Unlike static filtering which can only provide protection against a TCP SYN scan attack, dynamic packet filtering provides protection against TCP ACK scan as well as TCP FIN scan attacks. Dynamic packet filtering is also referred to as "intelligent filtering". This technique can also handle traffic-control effortlessly with UDP traffic, as it no longer relies on the UDP header for session-state information.

2.4.5 Stateful Filtering

Stateful filtering, an advanced filtering technique, tracks session context in addition to session state. Stateful filtering involves analyzing individual data packets in the traffic flow and monitoring information regarding the application protocol in addition to the contents of the

packet header and payload. The term 'stateful' signifies the ability to remember the connection status. Also known as "Stateful Multilevel Inspection", this allows protocol-specific monitoring of even connectionless protocols like UDP, NFS and RPC.

2.4.6 Proxy Filtering

Proxy filtering uses proxy servers or application gateways to act as mediators between the source and the destination systems. Since all communication between the two hosts is controlled, this provides a simple and effective defense against port abuse.

2.4.7 Plug Gateways

Plug gateways use stripped-down proxies that are not application-specific. They allow connectivity for specified ports and the traffic control is based on the principle of dynamic packet filtering.

2.4.8 Port Address Translation

Port address translation routes incoming packets to pre-defined private IP addresses on the local network based on the port number-service mapping. This strategy can be used to foil specific port-based attacks.

2.4.9 Traffic Management

Traffic management, also known as bandwidth management, helps in identifying traffic on a port-by-port basis and allows implementation of policies to detect and control port abuse. Not only can traffic be denied to specific ports but the acceptable volume of traffic can also be

specified for any port. Traffic management can also be achieved by regulating traffic flow on the basis of applications or protocols.

CHAPTER 3

BANDWIDTH MANAGEMENT

This section of the thesis is broadly divided into three subsections—Goals, Preliminaries, and Bandwidth Management. While the first subsection is a formal statement of the goals of this thesis, the second subsection deals with the infrastructure issues associated with the tools used for conducting the research. The third subsection describes the methodology followed and lessons learnt.

3.1 Goals

The goal of this thesis was to study bandwidth management techniques. This thesis describes the lessons learnt as part of this study. The task involved was divided into two subtasks with the following goals:

- (a) Identify and minimize the use of excessive bandwidth by non-critical and undesirable applications (application-adaptive strategy)
- (b) Monitor bandwidth use to minimize port abuse and flow of malicious data (rogue traffic on legitimate ports)

3.2 Preliminaries

This section, which deals with the preliminaries, is divided into three subsections—‘Background study’, ‘Data Collection Tools’ and ‘Monitoring Station’. While the first part gives a brief overview of the background study, the second part describes the various tools used for

data collection along with the rationale for their selection. The third part deals with the network configuration aspects of the traffic monitoring station and the PacketShaper settings.

3.2.1 Background study

The primary sources for the background study were the websites, <http://citeseer.nj.nec.com/cs> and <http://www.sans.org> and ACM and USENIX journals. Other frequently visited security related web sites included <http://www.neohapsis.com>, <http://www.dshield.org>, <http://www.insecure.org>, <http://www.securityfocus.com>, and <http://www.incidents.org>. While work has been done in the field of Quality of Service (QOS) and network bandwidth management based on traffic classification, no study was found that concentrated on application adaptive bandwidth management or that measured desirable vs. undesirable bandwidth in a university setting. Most professional organizations have a clearly defined set of “topics of professional interest”. However, a university setting is unique in the sense that given the wide variety of courses taught, the topics of “professional interest” cover almost everything, thus making it difficult to clearly classify traffic into ‘desirable’ and ‘undesirable’ parts based only on the topic content. Also, a typical university network is more prone to exploitation by students using P2P file swapping services and other applications that subject the network to a considerable risk of Trojans and other malicious code.

3.2.2 Data Collection Tools

Data collection for the current thesis involved the use of various tools such as the PacketShaper system, PacketPup, Network Probe, Ethereal, Active Ports and the Action Request System software. The selection of the tools was influenced by financial considerations. In the absence of any specially allotted funds, an effort was made to optimally use the existing

infrastructure. While PacketShaper and the Action Request System were part of ETSU's existing infrastructure, all the other tools used were freeware, thereby keeping the budgetary overheads to a bare minimum.

3.2.2.1 PacketShaper

Packeteer's PacketShaper system consists of a combination of hardware and software components. ETSU has the PacketShaper 2500 hardware console and the associated software version presently being used is 5.3. PacketShaper supports application-adaptive bandwidth management across enterprise WANs and can be used for controlling inbound as well as outbound traffic. As part of this thesis, the PacketShaper tool was used extensively to monitor and analyze bandwidth utilization on the campus network. The PacketShaper client, being web-based, is platform independent. PacketShaper is used actively by OIT for assuring quality of service to the network users. The policies are regularly reviewed since requirements often change with the discovery of new threats and vulnerabilities. Findings of this thesis served as an input parameter for the choice of strategy adopted while using the PacketShaper.

3.2.2.2 Active Ports

SmartLine Inc's Active Ports version 1.4 is a freeware tool designed for the Windows NT/2000/XP platform. It was used extensively to monitor all open TCP and UDP ports on the local host. Active Ports displays all active connections on the host along with the IP addresses and port numbers of both the connection endpoints. It also facilitates monitoring applications on the basis of that application's associated ports. This helps in identifying open ports associated with Trojans or other vulnerabilities and allows its user to terminate suspect processes.

3.2.2.3 PacketPup

Palisade systems' PacketPup version 2.2 is a limited-time freeware trial package. PacketPup is a bandwidth-monitoring tool that aids in detecting P-2-P file-sharing or streaming media traffic on the network. PacketPup's biggest drawback is its lack of support for controlling traffic. PacketPup was used to verify the classification-effectiveness of PacketShaper.

3.2.2.4 Network Probe 0.4

Network Probe version 0.4 from Object Planet is a freeware protocol analyzer and network traffic monitoring tool. Network Probe was used to identify and track rogue traffic in real-time, and to obtain information about the network interface cards (NICs) of the communicating hosts. A sample screen shot depicting real-time network analysis using Network Probe is shown in figure 3.1.

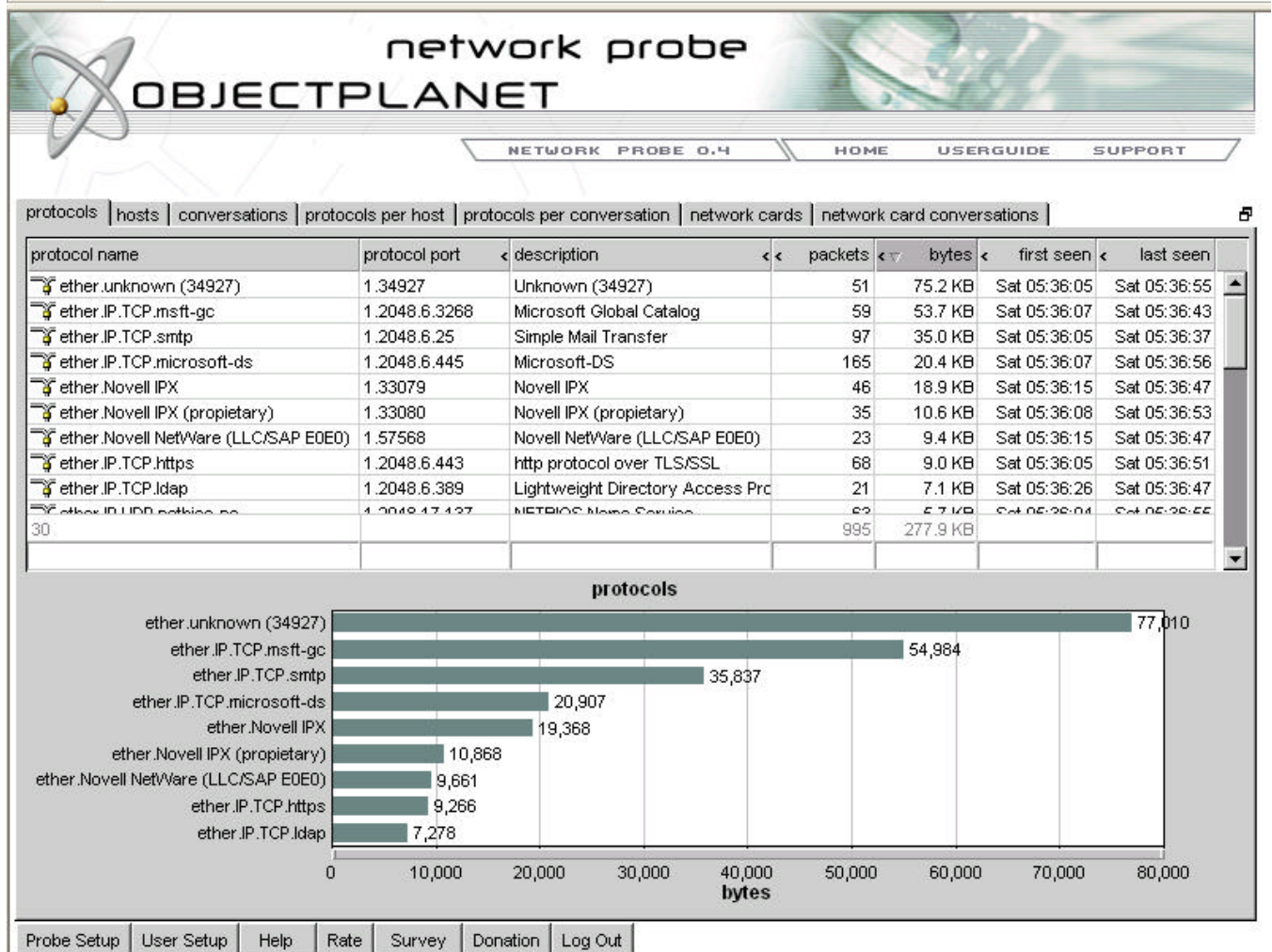


Figure 3.1: Network Probe in Action

3.2.2.5 Ethereal

Ethereal is a free network protocol analyzer for the Unix and Windows platforms. Ethereal version 0.9.7 was used to monitor network traffic in real time to identify rogue and undesirable traffic streams. Ethereal allows access to detailed traffic information on a packet-by-packet level. It was used extensively to reconstruct the traffic stream for different TCP sessions to allow a detailed investigation of the traffic contents. Ethereal's cost, together with its broad support for a wide variety of hardware and software platforms, cross-platform compatibility of data files and its powerful network analysis capabilities made it the protocol analyzer of choice

for this thesis. At present Ethereal claims to be able to successfully “dissect” 366 protocols. A detailed list of these protocols is given in Appendix A.

3.2.2.6 Action Request System

The Action Request System (ARS) is a Service Management utility marketed by Remedy Corporation. OIT uses ARS version 4.05.02 for managing its helpdesk services and inventory management functions. ARS was used in this work to help identify the various hosts on the campus network.

3.2.2.7 McAfee’s GroupShield Exchange 5.0 Enterprise Suite

GroupShield Exchange 5.0 Enterprise Suite consists of three components. The first, GroupShield, is a network based content filtering and anti-virus package with built in support for Microsoft’s Virus Scanning API. The second, ePolicy Orchestrator (ePO), facilitates centralized protection from malicious threats that include known exploits and malicious scripts aimed at subverting a network. The last, Outbreak Manager contains outbreaks of new viruses by monitoring network activity for malicious outbreak patterns.

3.2.3 Monitoring station

A monitoring station was set up for primary data collection at the OIT data center in Lucille Clement Hall. The station consisted of two computers at different topological positions on the ETSU network. One, a Windows 2000 platform, served as a network normal host. The other, a Redhat Linux 8.2 box positioned outside the firewall and the packet-shaper, afforded access to the entire incoming network traffic before it encountered any perimeter access control or security device. I had started with two different operating systems to allow flexibility in

selecting the best possible tools. This monitoring port is represented by a ‘star’ symbol as shown in Figure 3.2, depicting a part of the ETSU network’s topology.

A brief summary of PacketShaper settings is given below:

Table 3.1: Summary of PacketShaper settings

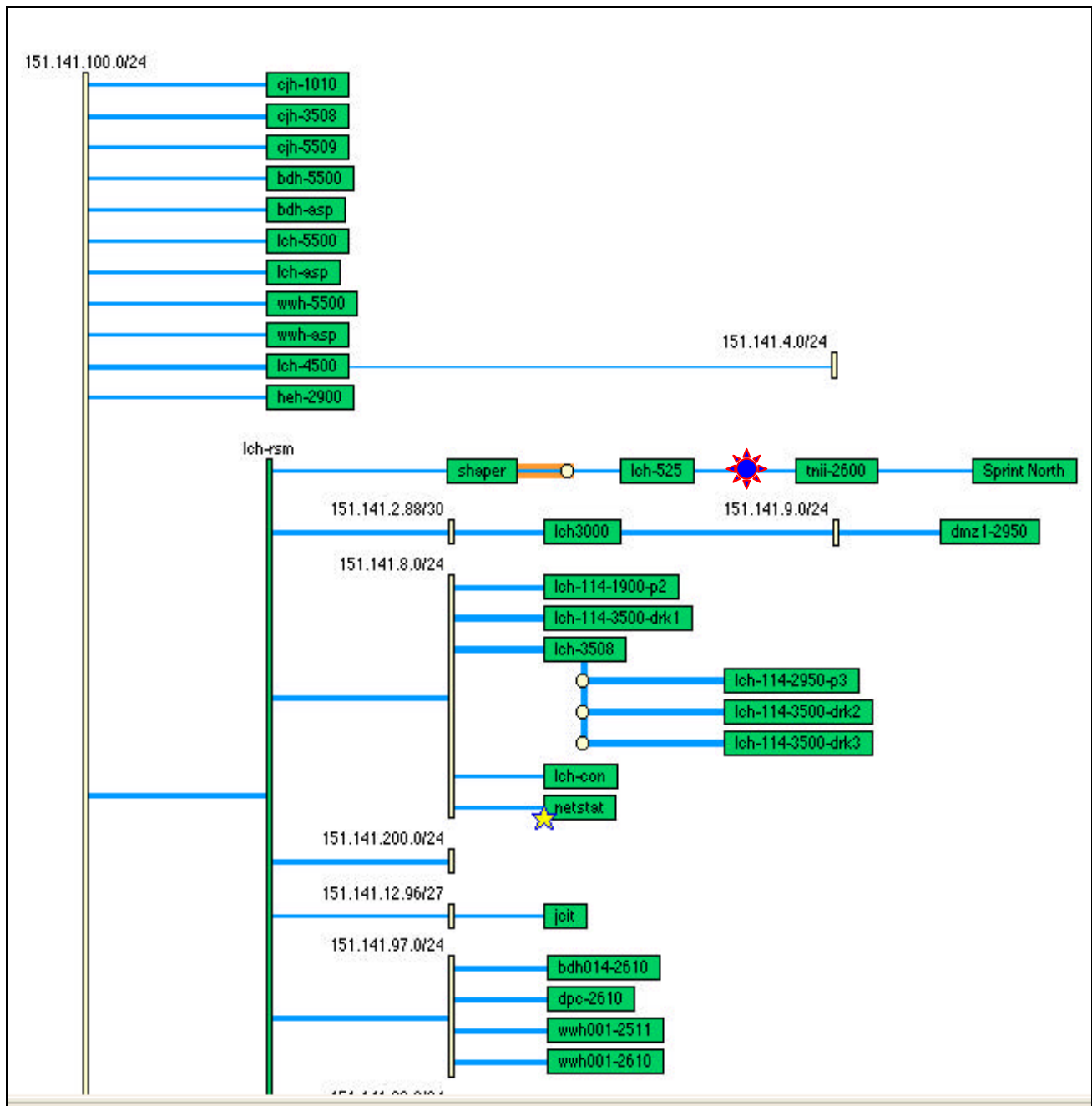
Non-sharable (local) settings:		Sharable settings:	
IP address:	151.141.95.3	Site router:	151.141.95.2
Subnet mask:	255.255.255.0	Link speed:	9264k
Gateway:	151.141.95.1	Packet shaping:	on
DNS server(s):	151.141.8.100	Traffic discovery:	on
Default domain:	etsu.edu	Automatic policy:	off
Inside nic speed:	100BaseT full-duplex		
Outside nic speed:	100BaseT full-duplex		

3.3 Bandwidth Management

Section 3.3 describes the methodology for achieving real-time bandwidth management. The process was divided into three distinct stages—classification of the network traffic, analysis and interpretation of the classified traffic, and enforcement of management policies.

3.3.1 Classification of the Network Traffic

A detailed analysis of the network traffic is a prerequisite for achieving efficient bandwidth management. To aid this analysis, the entire network traffic was classified in real time, using PacketShaper, into distinct categories or classes. The classification was achieved using 'matching rules' to identify different types of traffic in the network stream. Depending on requirements, the matching rules were configured to classify data on the basis of, port number, protocol family, application / service, and web attributes. When a certain traffic flow satisfied a particular matching rule, a corresponding entry was added to the classification tree. A typical hierarchical traffic tree depicting the various classes is shown in the figure 3.3.



Sprint North: ISP

TNII-2600: Connection to the Tennessee Information Infrastructure (service provider for Tennessee state organizations).

★ : Monitoring Station

LCH-525: Cisco PIX 525 Firewall

Shaper: Packeteer's PacketShaper 2500 Hardware console.

Figure 3.2: Position of the Monitoring Station

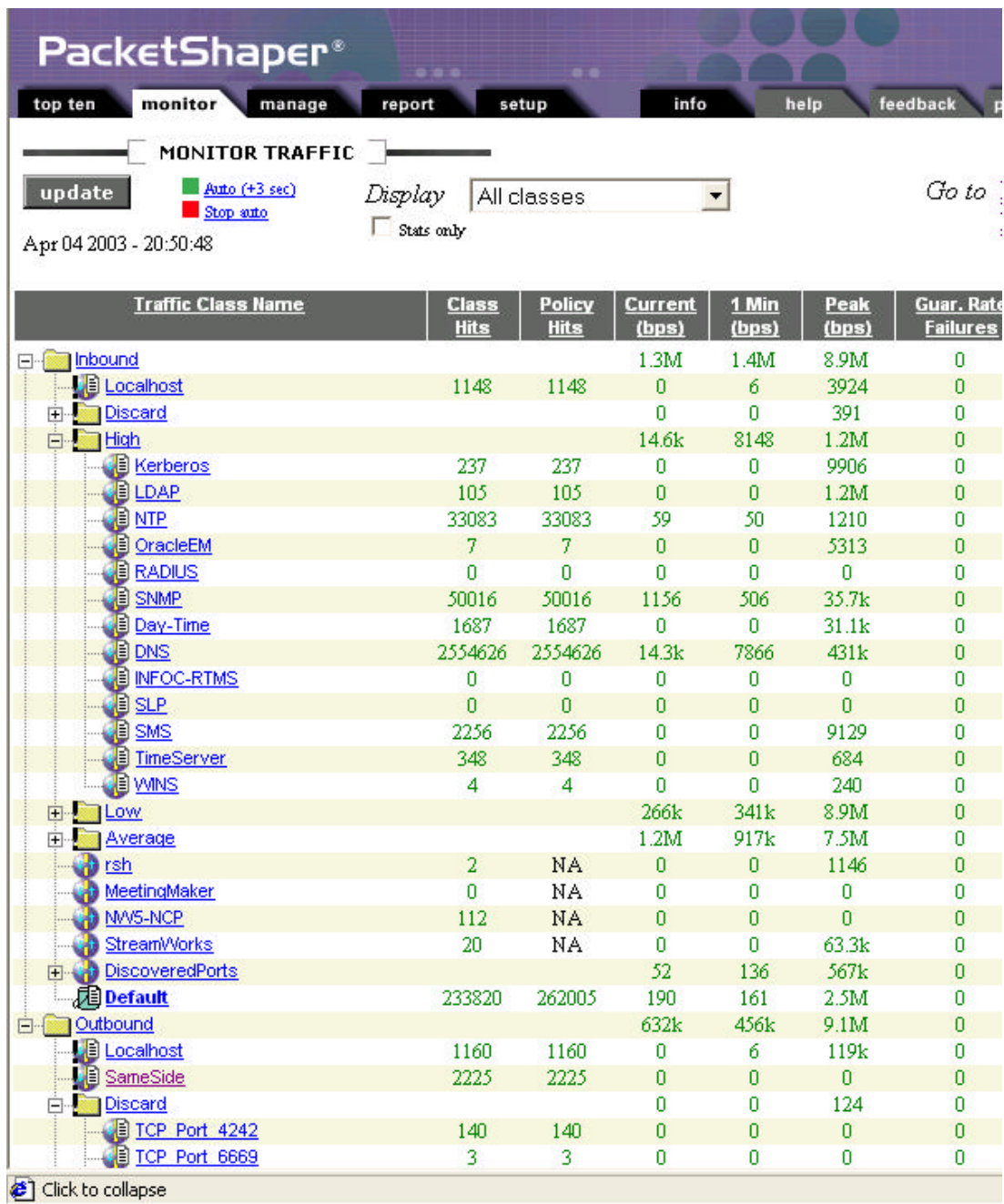


Figure 3.3: Typical Hierarchical Network Traffic Classification Tree

Port-based classification classifies Layer 4 traffic by port number or port number range. The use of ranges helps in identifying and restricting traffic directed at undesirable port numbers. It also helps in measuring the volume of traffic directed at commonly used well-known ports.

Protocol based classification classifies traffic by protocol type. PacketShaper³⁶ supports traffic classification based on transport protocol (viz. TCP and UDP) as well as protocol family, such as IP, SNA, Net BIOS, AppleTalk, and IPX etc. This classification was of tremendous value for managing bandwidth on the basis of protocols used.

Application-adaptive classification uses Layer 7 application signatures to identify a communication's participating applications. While protocol-based and port-based traffic classification is helpful for standard applications using static-port assignments, the strategy falls short for applications using non-standard ports or dynamically negotiated ports. This sort of tracking is supported by PacketShaper's ability to track traffic with migrating port-assignments, and to use application-specific identifier to differentiate among applications using the same port number. Examples of applications using dynamically negotiable ports include passive FTP, AOL instant messaging, and peer-to-peer file sharing applications like KaZaA, Gnutella, and Napster. Another example of application-adaptive classification for applications using the same port (port # 23) include differentiating traffic streams for TN3270 and TN5250 data from other Telnet traffic. Application-adaptive classification can also isolate different types of traffic for a single application based on different matching rules. This application sub-classification allows separation of Oracle netv2 traffic based on version (Oracle 7 Vs Oracle 8i/9i), as well as participating database. Similarly VoIP data can be sub-classified based on CODEC. Packeteer claims that PacketShaper can classify more than 150 types of traffic streams. A complete list of applications, protocols and services classified³⁷ by PacketShaper is placed in Appendix B.

Web Classification was motivated by the initial indication that Web-directed HTTP traffic consumed the bulk of ETSU's network bandwidth. In order to differentiate desirable and undesirable traffic, various identifying attributes such as IP addresses, subnets, URLs, server

location, mime type, HTTP tunnel, and traffic direction (inbound vs. outbound) were used for finer classification. This facilitated identification of mission-critical traffic from entertainment-oriented media traffic using HTTP. Streaming media data uses the real-time protocol (RTP), which uses various identifying attributes such as the media type (audio vs. video), the encoding name and the clock rate. With the help of these identifying attributes, a very fine classification of the network traffic is possible.

3.3.2 Analysis and Interpretation

An initial traffic analysis of the ETSU network, conducted in August 2002, was used to identify the top ten types of consumers of ETSU network bandwidth, relative to average flow, peak flow, and total volume of traffic. Network traffic trends were monitored on almost a daily (except weekends) basis. Any knowledge of new vulnerabilities or increased network transaction delays prompted an analysis of host-level bandwidth usage for the affected traffic class. The host analysis thus formed a secondary step used for precision-tuning the bandwidth allocation. The top talkers (originators of network traffic) and listeners (recipients of network traffic) were identified over different time intervals. Sample data showing the DNS name, IP address and percentage bandwidth usage for the top users of outbound FTP and the pcAnywhere remote monitoring application is shown in tables 3.2 through 3.5.

Table 3.2: Top sending IP hosts in class /Outbound/Low/FTP

Top Talkers			
	DNS Name	IP Address	Usage
1	infoserv	151.141.8.184	26%
2	ftp	151.141.8.164	11%
3	antivirus	151.141.8.167	5%
4	techweb	151.141.48.21	5%

5	webserv	151.141.8.154	3%
6	casanetn	151.141.8.172	3%
7	cscidbw	151.141.30.36	2%
8	ats	151.141.30.127	<1%
9	qcom	151.141.8.171	<1%
10	Server refused request	151.141.56.101	<1%

Table 3.3: Top receiving IP hosts in class /Outbound/Low/FTP

Top Listeners			
	DNS Name	IP Address	Usage
1	No such name	216.145.70.254	9%
2	ool-182f71d9.dyn.optonline.net	24.47.113.217	<1%
3	62-101-125-229.fastres.net	62.101.125.229	<1%
4	pcp03046746pcs.grey01.tn.comcast.net	68.62.247.7	<1%
5	No such name	205.227.136.41	<1%
6	jc-c-24-158-136-55.chartertn.net	24.158.136.55	<1%
7	hosting.web-axis.net	216.127.80.46	<1%
8	morristown-68-118-102-92.chartertn.net	68.118.102.92	<1%
9	No such name	161.69.201.238	<1%
10	No such name	161.69.201.237	<1%

Table 3.4: Top sending IP hosts in class /Outbound/Average/pcANYWHERE

Top Talkers			
	DNS Name	IP Address	Usage
1	yan	151.141.55.207	83%
2	krish	151.141.55.172	13%
3	housing	151.141.8.177	<1%
4	blackboard	151.141.8.51	<1%
5	einstein	151.141.30.144	<1%
6	Server refused request	151.141.28.215	<1%
7	etsu81240	151.141.55.159	<1%
8	etsu82375	151.141.60.233	<1%
9	wrl-voyager	151.141.112.103	<1%

Table 3.5: Top receiving IP hosts in class /Outbound/Average/pcANYWHERE

Top Listeners			
	DNS Name	IP Address	Usage
1	p50862209.dip0.t-ipconnect.de	80.134.34.9	<1%
2	acaen-105-1-6-114.abo.wanadoo.fr	81.48.27.114	<1%
3	host241-107.pool80117.interbusiness.it	80.117.107.241	<1%
4	pd9e3d083.dip.t-dialin.net	217.227.208.131	<1%
5	host157-174.pool80116.interbusiness.it	80.116.174.157	<1%
6	213-208-104-231.dyn.gotadsl.co.uk	213.208.104.231	<1%
7	pd9e7e1ec.dip.t-dialin.net	217.231.225.236	<1%
8	198.knoxville-05rh16rt-ca.dial-access.att.net	12.93.225.198	<1%
9	42.knoxville-04rh15rt-ca.dial-access.att.net	12.93.222.42	<1%
10	No such name	209.126.214.41	<1%

3.3.3 Enforcement of Management Policies

Management strategies used included implementation of application-based traffic flow policies. These policies allowed assured levels of bandwidth for desirable traffic, while restricting undesirable traffic flows. The various policy types employed were priority, flow rate, discard, and ignore.

Priority policies were used to assign priority levels for various traffic classes. PacketShaper priorities range from 0 (minimum priority) to 7 (maximum priority) with a default of 3. Assigning priority policies are most suitable for non-continuous or short, unpredictable, non-IP traffic such as RADIUS authentication, Telnet, and DNS queries.

Flow rate policies were used to assign a minimum acceptable flow-rate for specific traffic classes. Using the 'burstable at priority' feature of flow rate policies grants applications higher-than-minimum assigned bandwidth subject to availability. If all assigned classes already have their minimum flow rate and more bandwidth is available, then individual class-priority becomes the deciding factor for distribution of excess bandwidth. Flow rate policies also allow the setting of a limit on the maximum acceptable bandwidth for different traffic classes. Flow rate policies are most suitable for bursty IP-based traffic such as FTP and HTTP as well as for

latency-sensitive applications such as Voice over IP. These policies were implemented using PacketShaper's proprietary TCP rate-control technology.

The discard policy was used to discard all packets from pre-identified, non-desirable traffic classes. This policy must be applied with care to managing session-oriented TCP, since the resulting time-outs will delay feedback to the user. An alternative policy, *never-admit*, enforces flow control at the entry point while informing the user immediately whenever certain traffic is blocked. Traffic classes that are presently being blocked on the ETSU network using 'discard' include TCP_Port_4242, TCP_Port_6669, Audiogalaxy, CUSeeMe, Doom, eDonkey, Gnutella, IRC, KaZaA, Napster, Net2Phone, Quake, Unreal, Webshots, YahooGames, Battle.net, Blubster, Mythic, and SonyOnline.

The ignore policy classifies traffic that should be ignored, but nevertheless measured. 'Ignore' traffic is typically pass-through traffic that does not affect the bandwidth at routers and other access points of interest for a particular network. On the ETSU network, currently outbound DHCP and RIP traffic is being ignored.

In addition to these five policies, partitions were used to allocate bandwidth among the competing tasks. A partition is a virtual-pipe-like mechanism that can assure a pre-defined bandwidth capacity for any particular traffic class. Like policies, partitions assure minimum required bandwidth for desirable traffic classes while restricting bandwidth for undesirable classes. Unlike policies, partitions are typically used for aggregate traffic classes.

Partitions can also be fixed or burstable. While a fixed partition always allows a pre-defined amount of bandwidth, a burstable partition allows a class to use any excess bandwidth available over and above the pre-defined amount of bandwidth set for that class. The ETSU network uses a burstable partition of 2kbps size for inbound as well as outbound ICMP traffic:

irrespective of the amount of network traffic, a minimum of 2kbps of bandwidth is always set aside specifically for ICMP. Monitoring ICMP traffic is of utmost importance as many denial of service (DOS) attacks extensively use ICMP services.

Table 3.6 shows the partition summary for ICMP traffic as implemented on the ETSU network.

Table 3.6: Partition Summary for ICMP Traffic

Partition			Dynamic sub partition			
Name	Size-Limit	Current Guar./Excess	Size-Limit	Current Active/Idle	Max	Overflow
/Inbound	Uncommitted-none	0/9.2M	-	-	-	-
/Inbound/Average/ICMP	2000-none	0/0	-	-	-	-
/Inbound Max Total	2k-none	0/0	-	0/0	0	-
/Outbound	Uncommitted-none	0/9.1M	-	-	-	-
/Outbound/Average/ICMP	2000-none	0/0	-	-	-	-
/Outbound Max Total	2k-none	0/0	-	0/0	0	-

3.3.3.1 Feedback-Oriented Fine-Tuning

Subsequent to the implementation of various policies and partitions, ETSU network bandwidth utilization was analyzed by studying various parameters such as the number of class hits, the number of policy hits, and the current as well as peak values for the rate of flow of traffic (in bps) for different traffic classes. A typical screen shot used for monitoring these parameters is shown in the Figure 3.4.

Traffic Class Name	Class Hits	Policy Hits	Current (bps)	1 Min (bps)	Peak (bps)	Guar. Rate Failures	Partition Min-Max	Policy Type (Pri.) Guar.-Lin
Inbound			2.1M	1.9M	9.2M	0	uncommitted-none	
Localhost	4722	4722	9	13	2779	0		Priority (6)
Discard			0	0	279	0		
TCP Port 4242	394	394	0	0	0	0		Discard
TCP Port 6669	73	73	0	0	0	0		Discard
AudioGalaxy	1333	1333	0	0	0	0		Discard
CUSeeMe	3	3	0	0	0	0		Discard
Doom	15	15	0	0	0	0		Discard
eDonkey	55397	55397	0	0	4	0		Discard
Gnutella	10345	10345	0	0	19	0		Discard
Half-Life	0	0	0	0	0	0		Discard
IRC	895	895	0	0	279	0		Discard
KaZaA	127453	127453	0	0	32	0		Discard
Napster	1951	1951	0	0	0	0		Discard
Net2Phone	0	0	0	0	0	0		Discard
Quake	3	3	0	0	0	0		Discard
SHARESUDP	9	1	0	0	8	0		Discard
FTTP	3	0	0	0	0	0		Discard
Unreal	1	1	0	0	0	0		Discard
Webshots	76	76	0	0	181	0		Discard
YahooGames	649	649	0	0	0	0		Discard
Battle.net	0	0	0	0	0	0		Discard
Blubster	1	1	0	0	0	0		Discard
CU-DEV	2	0	0	0	0	0		Discard
Echo	225	427	0	0	2713	0		Discard
Mythic	0	0	0	0	0	0		Discard
SonyOnline	12	12	0	0	0	0		Discard
High			21.7k	20.8k	1.3M	0		
Kerberos	1151	1151	0	0	12.3k	0		Priority (6)
LDAP	464	464	1	0	1.3M	0		Priority (6)
NTP	130220	130220	29	39	1550	0		Priority (6)
OracleEM	30	30	0	0	563k	0		Priority (6)
RADIUS	5	5	0	0	0	0		Priority (6)
SNMP	152998	152998	2076	508	35.7k	0		Priority (6)
Day-Time	5627	5627	0	0	57.5k	0		Priority (6)
DNS	8472958	8472958	4028	3162	887k	0		Priority (6)
NTPC BTMS	5	5	17.5k	18.2k	71.6k	0		Priority (6)

Figure 3.4: Class Hits, Policy Hits and Traffic Volume for Different Classes

All traffic that generated interesting parametric values was then studied in depth. The shaping policy effectiveness was studied. The criteria used were comparison of expected and actual outcome for a particular traffic class on the basis of its associated policy. For example, it was found that a particular faculty host machine was persistently connected to the Kazaa file-swapping service for more than 36 hours in a particular session. The traffic measurements indicated that the entire traffic for this class was being discarded before it reached the host. Since the discard policy had been associated with Kazaa, this actual measurement was consistent the expected outcome and indicated that this particular policy implementation achieved its goal. Similarly, if the flow rate policy were assigned to a particular traffic class, the shaping

effectiveness would be determined by measuring whether the minimum flow rate as specified by the policy was indeed available for that traffic class. Other parameters examined included network efficiency, average and peak rate, class and partition utilization, utilization with peaks, transaction delays, response time, and bytes transmitted for a particular class, partition or a link. This data was collected over various time intervals (hourly, daily, weekly, and monthly) depending on the required precision of the sampling frequency and the total observation time-period for any particular class. An observation interval range of 1 to 7 days will have a sampling frequency of one hour whereas the same would be one minute for an observation interval of less than 24 hours. Sample screenshots shown in Figures 3.5 and 3.6 depict the outbound as well as inbound traffic patterns for average and peak flow rates for a typical month during fall semester, 2002. Hourly, daily, and weekly graphs for the corresponding period are in Appendix C.

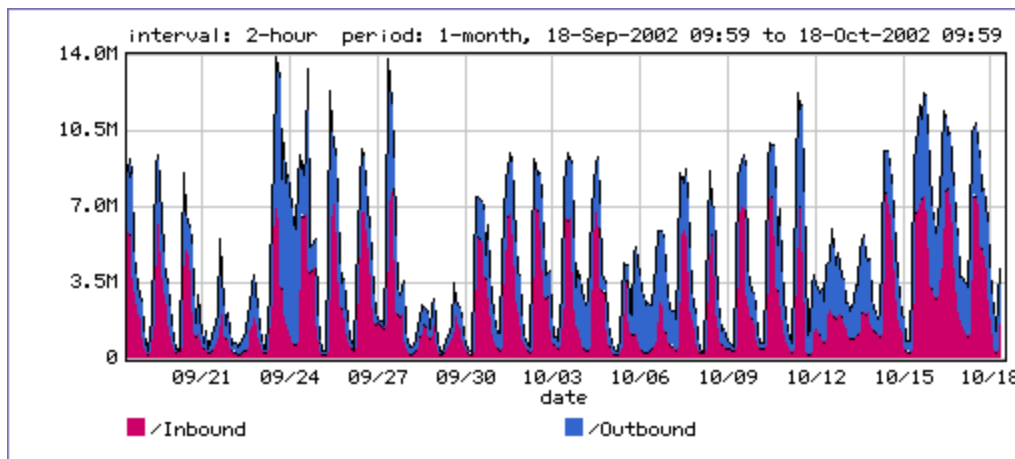


Figure 3.5: Monthly Average Rate (from 18 Sep 2002 to 18 Oct 2002)

The distribution curve in Figure 3.5 indicates a dip in the average traffic rate for both inbound as well as outbound traffic on Sep 21, 22, 28, and 29, and on Oct 5, 6, 12, and 13, 2002. All these dates correspond to weekends and the decreased flow rate is thus easily explained.

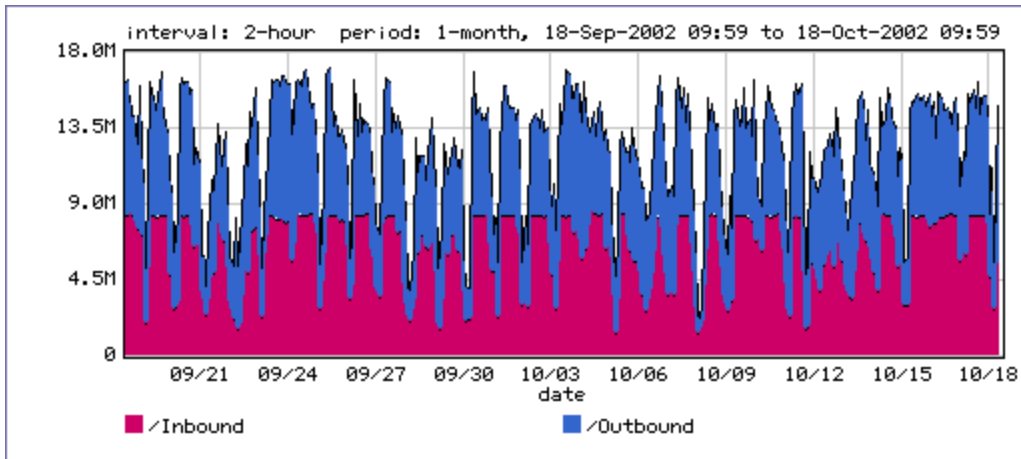


Figure 3.6: Monthly Peak Rate (from 18 Sep 2002 to 18 Oct 2002)

The distribution curve in Figure 3.6 shows less variance throughout the entire month. This is an indication of the stability of the packet shaping wherein the peak flow rates are maintained at an almost similar level irrespective of the actual volume of traffic encountered.

The inbound as well as outbound network utilization and efficiency curves along with the top ten classes for 17-18 Oct 2002 are shown below in Figures 3.7 through 3.12. During this time period (24 hours), the total inbound traffic was 44917399 Kbytes whereas the total outbound traffic was 21853781 Kbytes. The corresponding data for hourly, weekly and monthly time intervals is in Appendix D.

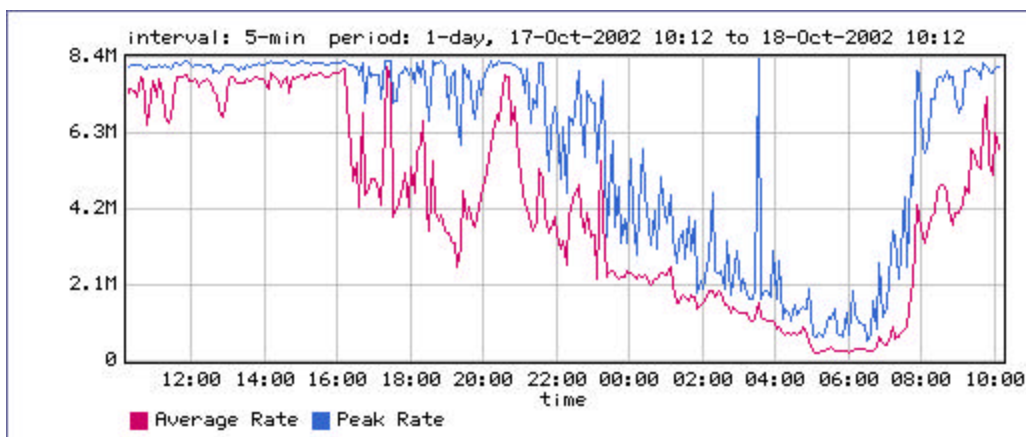


Figure 3.7: Inbound Utilization (from 17 Oct 2002 to 18 Oct 2002)

The inbound link utilization as shown above indicates the lean period for the average rate (in bits per second) to be from midnight to 8:00 AM. This was expected.

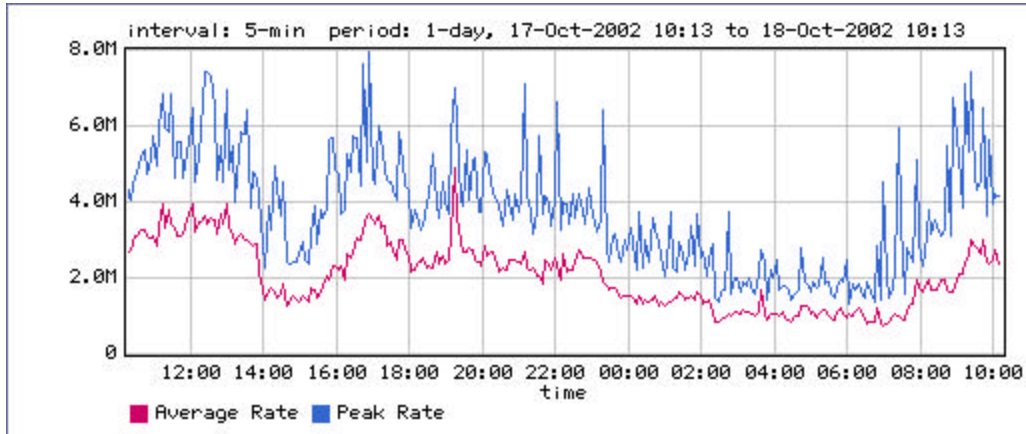


Figure 3.8: Outbound Utilization (from 17 Oct 2002 to 18 Oct 2002)

Unlike for inbound utilization, the outbound network utilization curve indicates that the severity of the dips is much less. The average rate rises around 7:00PM, when the campus computer labs have most students. Also, the relative dip in the average outbound rate is predominant from 2:00 AM to 8:00 AM—the exact time when the student computer labs are shut.

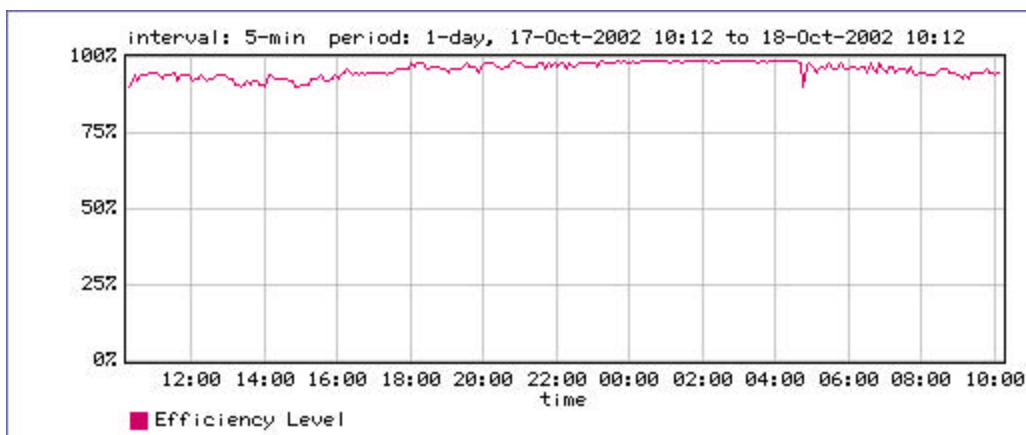


Figure 3.9: Inbound Network Efficiency (from 17 Oct 2002 to 18 Oct 2002)

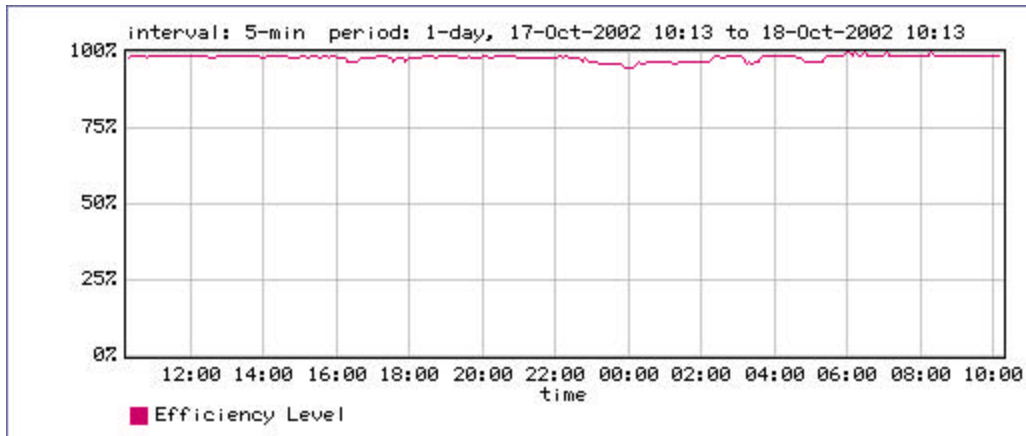


Figure 3.10: Outbound Network Efficiency (from 17 Oct 2002 to 18 Oct 2002)

The network efficiency curves show the percentage of packets that are properly transmitted in the first attempt itself. The dips in the efficiency curve indicate retransmissions as well as lost packets. The efficiency level is calculated as a value = (Total bytes - Retransmitted bytes) / Total bytes and then expressed in percentage. These curves can thus be used as broad indicators of the network's health.

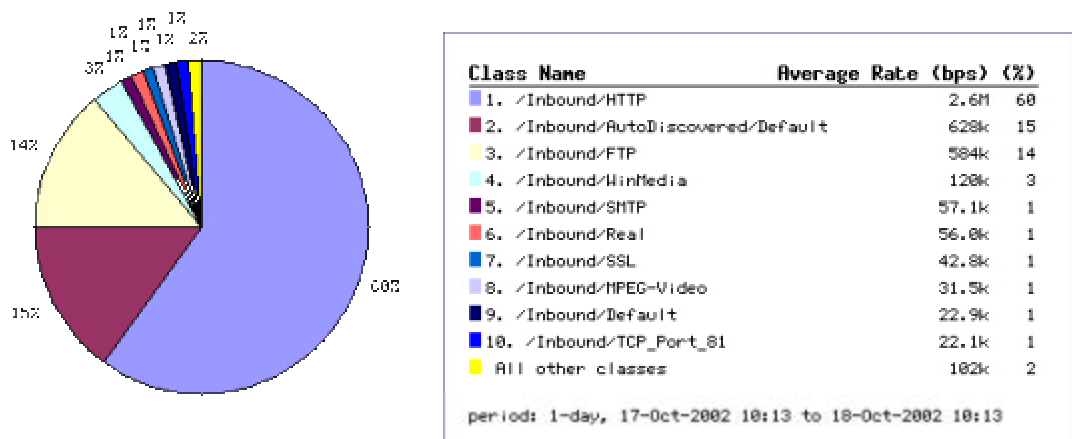


Figure 3.11: Inbound Top 10 Classes (from 17 Oct 2002 to 18 Oct 2002)

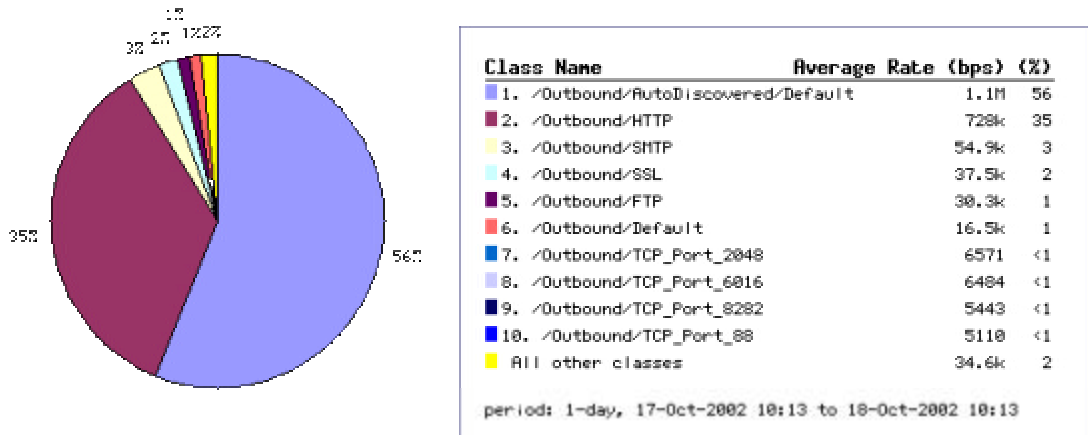


Figure 3.12: Outbound Top 10 Classes (from 17 Oct 2002 to 18 Oct 2002)

For the 24-hour period under consideration, the top ten inbound and outbound classes by percentage volume of traffic and their average flow rate in bits per second is shown in Figures 3.11 and 3.12.

The application response time data for the outbound AOL-IM-ICQ traffic class based on 24 hour time period from 24 Oct 2002 to 25 Oct 2002 can be studied using the transaction delay curves as well as the histograms depicting the distribution of the transaction delay for that class. These two graphs are shown in the figures 3.13 and 3.14.

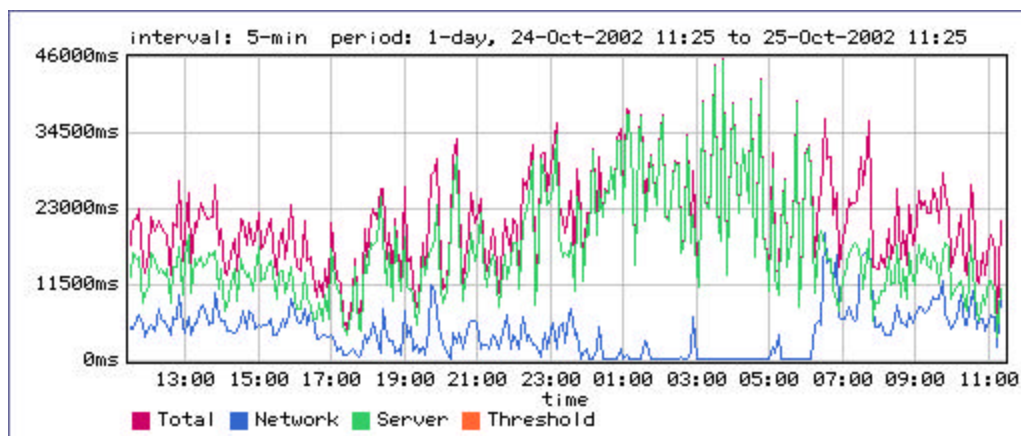


Figure 3.13: Transaction Delay

The graph in Figure 3.13 represents a timeline of the average response time (in ms) for the outbound AOL-IM-ICQ traffic class over a 24-hour period. Pink, blue and green lines indicate the total, network and server transaction delays. Any sudden or drastic increase in the transaction delays for traffic classes indicates possible bandwidth congestion on that link. The sampling frequency used in this graph is 5 minutes.

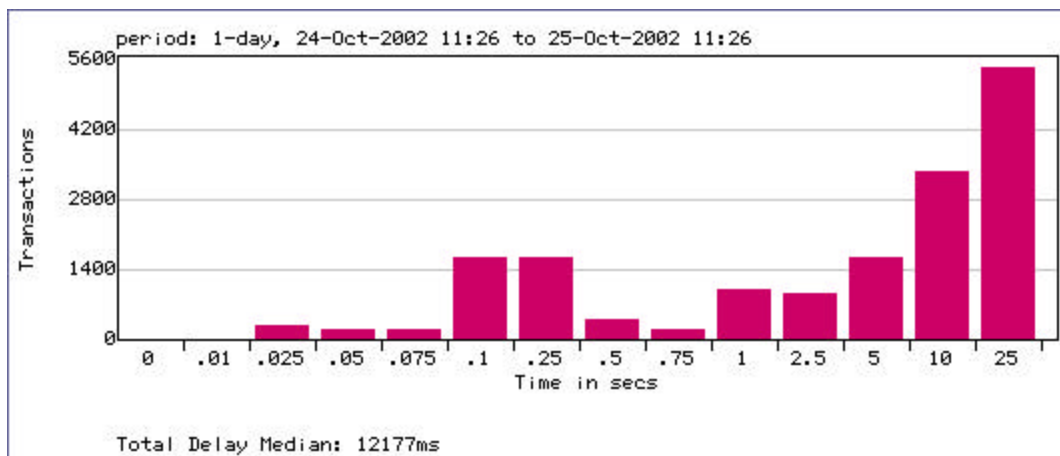


Figure 3.14: Transaction Delay Distribution

The transaction delay distribution histogram is based on 14 distinct categories (ranging from 0 to 25 seconds) of response times plotted against the frequency of transaction delays in each category. The median delay value (12177 ms) is indicative of the actual transaction delay experienced by most network hosts. This use of a median value, as opposed to an average value, gives a more realistic indication in the event of a few extreme values.

Inputs based on traffic monitoring and other relevant data such as discovery of new vulnerabilities or threats were used to tweak policies for all desirable traffic classes until a minimum of 99% flow rate efficiency was achieved. The detailed traffic tree as on 01 April 2003 is given in Appendix E.

3.3.4 Data Integrity Verification

PacketShaper data was verified using a second, independent utility, PacketPup. Verification involved checking whether PacketShaper correctly identified bandwidth usage on a class-by-class basis. Discrepancies between the tools' classifications were attributable to PacketShaper's support for finer degrees of traffic classification (see Appendix F for a detailed list of protocols and services recognized by PacketPup). For example, PacketShaper divides SNMP traffic, which PacketPup classifies as one stream, into distinct classes such as SNMP Mon (monitor) and SNMP Trap (traps). PacketShaper can also subclassify the traffic for the game "Quake" depending on the transport protocol used (TCP or UDP) or the game's version. A sample screenshot depicting the parsing of the various rules by PacketPup is shown in figure 3.15. Sample log file data showing the identification of various protocols in the network traffic using PacketPup is placed in Appendix G.

3.3.5 Non-Standard Traffic

Neither PacketShaper nor PacketPup detects non-standard, rogue traffic in standard traffic flow. This rogue traffic includes non-standard traffic on standard ports that cannot be classified further on the basis of standard traffic-identifying attributes. Identification of non-standard traffic in standard channels of data exchange is important as there are certain ports that cannot be blocked even if they are known Trojan ports, since they are required for certain basic standard services. A step-by-step analysis of sample data is given below to illustrate the use of Network Probe 0.4 for identifying rogue traffic directed at standard ports. This analysis was carried on a 'Friday night-early Saturday morning' session as that is one of the most preferred times for students on the campus network to generate undesirable traffic.



Figure 3.15: PacketPup in Action

3.3.5.1 Step I: Initiate Network Probe

This step involved configuring the network probe application and the local client to monitor data flow on a given network. This data was tabulated as shown below to divide the traffic stream on the basis of protocols. The details include the protocol's name, the IANA assigned number for the protocol-port combination, the protocol's description, the number of packets transmitted, the number of bytes in the traffic and each flow's start and end time. An example for interpreting the protocol port column is depicted by selecting the corresponding value from the first row. Here, the number 1.2048.6.443 denotes the following:

- 1 == Ethernet
- 2048 == (0x0800) or IP
- 6 == TCP
- 443 == the port number for the HTTPS protocol (HTTP over TLS/SSL)

Table 3.7 below shows the entire traffic at the monitoring point.

Table 3.7: Network Traffic Flow Data

Protocol Name	Protocol Port	Description	Packets	Bytes	First Seen	Last Seen
ether.IP.TCP.https	1.2048.6.443	http protocol over TLS/SSL	2868	366946	Fri 23:34:23	Fri 23:42:34
ether.Novell IPX	1.33079	Novell IPX	813	361167	Fri 23:34:23	Fri 23:42:29
ether.Novell NetWare (LLC/SAP E0E0)	1.57568	Novell NetWare (LLC/SAP E0E0)	406	179933	Fri 23:34:23	Fri 23:42:29
ether.unknown (34927)	1.34927	Unknown (34927)	490	739900	Fri 23:34:23	Fri 23:42:33
ether.ARP	1.2054	ARP	541	32442	Fri 23:34:23	Fri 23:42:31
ether.IP.TCP.pop3s	1.2048.6.995	pop3 protocol over TLS/SSL (was spop3)	224	15334	Fri 23:34:23	Fri 23:41:54
ether.IP.TCP.smtp	1.2048.6.25	Simple Mail Transfer	10143	11120362	Fri 23:34:23	Fri 23:42:32
ether.IP.UDP.kerberos	1.2048.17.88	Kerberos	143	192010	Fri 23:34:25	Fri 23:42:32
ether.IP.TCP.idap	1.2048.6.389	Lightweight Directory Access Protocol	270	173354	Fri 23:34:25	Fri 23:42:32
ether.PCS Basic Block Protocol	1.16962	PCS Basic Block Protocol	245	14700	Fri 23:34:24	Fri 23:42:33
ether.IP.TCP.msftgc	1.2048.6.3268	Microsoft Global Catalog	465	462637	Fri 23:34:24	Fri 23:42:32
ether.IP.UDP.idap	1.2048.17.389	Lightweight Directory Access Protocol	8	1642	Fri 23:34:24	Fri 23:41:02
ether.IP.UDP.netbios-ns	1.2048.17.137	NETBIOS Name Service	136	12680	Fri 23:34:24	Fri 23:42:33
ether.IP.ICMP	1.2048.1	Internet Control Message [RFC792]	170	8793	Fri 23:34:24	Fri 23:42:24

ether.IP.TCP.pop3	1.2048.6.110	Post Office Protocol - Version 3	94	41796	Fri 23:34:24	Fri 23:41:49
ether.unknown (34925)	1.34925	Unknown (34925)	922	55320	Fri 23:34:24	Fri 23:42:34
ether.IP.TCP.domain	1.2048.6.53	Domain Name Server	260	19303	Fri 23:34:31	Fri 23:42:32
ether.IP.UDP.bootps	1.2048.17.67	Bootstrap Protocol Server	54	20244	Fri 23:34:30	Fri 23:42:31
ether.IP.ESP	1.2048.50	Encapsulating Security Payload [RFC1827]	3181	4429190	Fri 23:34:29	Fri 23:42:33
ether.IP.UDP.anarchy-online	1.2048.17.7500	Anarchy Online	31	3726	Fri 23:34:28	Fri 23:41:58
ether.IP.OSPFIGP	1.2048.89	OSPFIGP [RFC1583,JTM4]	44	3432	Fri 23:34:27	Fri 23:42:25
ether.IP.UDP.netbios-dgm	1.2048.17.138	NETBIOS Datagram Service	62	15084	Fri 23:34:26	Fri 23:42:13
ether.Novell IPX (proprietary)	1.3308	Novell IPX (proprietary)	533	197750	Fri 23:34:25	Fri 23:42:33
ether.IP.EIGRP	1.2048.88	EIGRP [CISCO,GXS]	212	15688	Fri 23:34:25	Fri 23:42:34
ether.DEC LAT	1.2458	DEC LAT	16	1872	Fri 23:34:45	Fri 23:42:19
ether.IP.TCP.nntp	1.2048.6.119	Network News Transfer Protocol	48	3456	Fri 23:34:41	Fri 23:41:41
ether.IP.UDP.mdns	1.2048.17.5353	Multicast DNS	1	130	Fri 23:34:35	Fri 23:34:35
ether.IP.TCP.www-http	1.2048.6.80	World Wide Web HTTP	247	28563	Fri 23:34:32	Fri 23:42:32
ether.IP.TCP.imaps	1.2048.6.993	imap4 protocol over TLS/SSL	199	18690	Fri 23:34:32	Fri 23:42:32
ether.unknown (417)	1.417	Unknown (417)	25	1500	Fri 23:34:55	Fri 23:42:15
ether.unknown (418)	1.418	Unknown (418)	25	1500	Fri 23:34:55	Fri 23:42:15
ether.IP.TCP.msxch-routing	1.2048.6.691	MS Exchange Routing	2	108	Fri 23:34:54	Fri 23:39:54
ether.IP.UDP.svrlc	1.2048.17.427	Server Location	8	728	Fri 23:35:08	Fri 23:35:40
ether.IP.UDP.domain	1.2048.17.53	Domain Name Server	22	2806	Fri 23:35:00	Fri 23:42:30
ether.IP.UDP.snmp	1.2048.17.161	SNMP	24	6976	Fri 23:35:22	Fri 23:42:22
ether.unknown (8192)	1.8192	Unknown (8192)	8	2904	Fri 23:35:17	Fri 23:42:17
ether.IP.TCP.imap	1.2048.6.143	Internet Message Access Protocol	110	12800	Fri 23:35:13	Fri 23:42:13
ether.IP.TCP.netbios-ssn	1.2048.6.139	NETBIOS Session Service	47	43165	Fri 23:36:27	Fri 23:38:18
ether.IP.TCP.microsoft-ds	1.2048.6.445	Microsoft-DS	132	26562	Fri 23:36:27	Fri 23:42:20
ether.IP.TCP.trellisagt	1.2048.6.2077	TrelliSoft Agent	21	5639	Fri 23:36:27	Fri 23:36:27
ether.IP.TCP.epmap	1.2048.6.135	DCE endpoint resolution	16	1540	Fri 23:36:27	Fri 23:36:46
ether.IP.TCP.unknown	1.2048.6.-1	Unknown	10	5924	Fri 23:36:46	Fri 23:40:05
ether.DEC MOP Remote Console	1.24578	DEC MOP Remote Console	2	2298	Fri 23:36:55	Fri 23:38:53
ether.IP.UDP.unknown (42342)	1.2048.17.42342	Unknown (42342)	1	118	Fri 23:39:00	Fri 23:39:00
ether.IP.UDP.isakmp	1.2048.17.500	isakmp	3	482	Fri 23:41:23	Fri 23:41:26
ether.IP.UDP.bootpc	1.2048.17.68	Bootstrap Protocol Client	4	1368	Fri 23:41:01	Fri 23:41:25

A study of the above traffic shows incoming traffic on port 119 (as highlighted above) using the Network News Transfer Protocol. As listed in the websites mentioned in section 2.1.1.2.1, port 119 is a known target port of the Trojan ‘Happy99’. According to the CERT® Incident Note³⁸ IN-99-02, the ‘Happy99’ Trojan is known to have a number of aliases such as

SKA, Win32.ska.a, Ska.exe, Wsock32.ska, I-worm.Happy, PE_SKA, Happy, and W32/Skanew and affects the Microsoft Windows family of Operating Systems. The worm includes the files³⁹ Happy99.exe (or Happy00.exe) of size exactly 10,000 bytes, Ska.exe, Ska.dll, Wsock32.dll, and Liste.ska. Happy99.exe displays fireworks⁴⁰ in a window titled "Happy New Year 1999" (cf. Figure 3.16), while infecting WSOCK32.dll and modifying the registry in the background. Once the system is infected, a message with the subject 'HAPPY99' or 'HAPPY00' and with a uuencoded attachment of the Happy99.exe file is sent to all email and Usenet addresses that receive mail from the infected system. The file 'liste.ska' keeps track of all the addresses to which the Trojan is sent from a particular host. Other symptoms⁴¹ of Happy99 infection include the possibility of the following error messages encountered while attempting to transmit information on the Internet:

- (i) "Outlook caused an Invalid Page Fault in module Unknown"
- (ii) "Explorer caused an invalid page fault in module Mailnews.dll at 014f: 62060a0f"
- (iii) "MSIMN caused an invalid page fault in module unknown"
- (iv) "MSIMN caused an invalid page fault in module Inetcomm.dll"
- (v) "MSIMN caused an invalid page fault in module Kernel32.dll"



Fig 3.16: Execution of the Happy99 Trojan

3.3.5.2 Step II: Isolate Communication for Interesting Ports

The next step of the investigation identified ETSU hosts involved in conversations using NNTP. Host usage of port 119 was tabulated for all hosts actively using this port, as shown in Table 3.8. This table shows NNTP usage by mail.etsu.edu and netstat.etsu.edu. Mail.etsu.edu is an ETSU campus mail server that primarily services faculty and staff email traffic. Netstat.etsu.edu, an InterMapper server, provides a graphical representation of real-time traffic flow and aides in monitoring the health of the campus network.

Table 3.8: Hosts using NNTP

Host Name	Address	Protocol	Packets In	Bytes In	Packets Out	Bytes Out	First Seen	Last Seen
mail.etsu.edu	151.141.8.105	ether.IP.TCP.nntp	94	8060	0	0	Fri 23:34:41	Fri 23:34:41
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.nntp	0	0	94	8060	Fri 23:34:41	Fri 23:48:41

The information in the ‘Packets/bytes in’ and ‘Packets/bytes out’ indicates that netstat.etsu.edu is the source and mail.etsu.edu is the destination host.

3.3.5.3 Step III: Identification of Potential Secondary Target Ports

While Happy99 uses port 119, one of the versions uses port 25 if port 119 is unavailable. Port 25 is officially assigned to the Simple Mail Transfer Protocol for both TCP as well as UDP. In order to make a detailed analysis of the traffic flow and to rule out rogue data directed at secondary target ports, data was collected giving details of all the protocols used by the source as well as the destination hosts. This is tabulated in tables 3.9 and 3.10 respectively. This additional information (as highlighted below) was used to monitor traffic associated with SMTP.

Table 3.9: All Protocols used by source host (netstat.etsu.edu)

Host Name	Address	Protocol	Packets In	Bytes In	Packets Out	Bytes Out	First Seen	Last Seen
netstat.etsu.edu	151.141.8.57	ether.IP.UDP.snmp	0	0	72	20723	Sat 00:00:23	Sat 00:20:24
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.pop3s	0	0	84	6888	Sat 00:00:41	Sat 00:20:41
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.nntp	0	0	131	11236	Sat 00:00:41	Sat 00:20:41
netstat.etsu.edu	151.141.8.57	ether.IP.UDP.bootps	0	0	21	12684	Sat 00:00:41	Sat 00:20:41
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.imaps	0	0	211	24923	Sat 00:00:32	Sat 00:20:32
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.smtp	0	0	152	13252	Sat 00:00:32	Sat 00:20:34
netstat.etsu.edu	151.141.8.57	ether.IP.TCP.www-http	0	0	345	26684	Sat 00:00:04	Sat 00:20:32
netstat.etsu.edu	151.141.8.57	ether.IP.UDP.sviloc	0	0	12	1260	Sat 00:05:32	Sat 00:20:53
netstat.etsu.edu	151.141.8.57	ether.ARP	0	0	38	2280	Sat 00:03:17	Sat 00:19:59

Table 3.10: All Protocols used by destination host (mail.etsu.edu)

Host Name	Address	Protocol	Packets In	Bytes In	Packets Out	Bytes Out	First Seen	Last Seen
mail.etsu.edu	151.141.8.105	ether.IP.TCP.pop3s	407	35009	0	0	Sat 00:00:05	Sat 00:00:05
mail.etsu.edu	151.141.8.105	ether.IP.TCP.nntp	145	12428	0	0	Sat 00:00:41	Sat 00:00:41
mail.etsu.edu	151.141.8.105	ether.IP.TCP.smtp	12217	14141882	0	0	Sat 00:00:06	Sat 00:00:06
mail.etsu.edu	151.141.8.105	ether.IP.TCP.https	4294	610157	0	0	Sat 00:00:04	Sat 00:00:04
mail.etsu.edu	151.141.8.105	ether.IP.TCP.imaps	500	54406	0	0	Sat 00:00:32	Sat 00:00:32
mail.etsu.edu	151.141.8.105	ether.IP.TCP.www-http	465	38864	0	0	Sat 00:00:04	Sat 00:00:04
mail.etsu.edu	151.141.8.105	ether.ARP	5	300	0	0	Sat 00:04:23	Sat 00:04:23
mail.etsu.edu	151.141.8.105	ether.IP.ICMP	2	1612	0	0	Sat 00:09:25	Sat 00:09:25

3.3.5.4 Step IV: Isolating Conversations from ‘Suspect’ Hosts

This step involved isolating all possible conversations originating from the ‘suspect’ host. This step gives all the destination hosts for the entire traffic from the source under investigation. It also helps in minimizing damage to the network when the source host is transmitting malicious data. This data is tabulated below in table 3.11.

Table 3.11: Conversations originating from netstat.etsu.edu

Source Host	Source Address	Destination Host	Dest Address	Packets	Bytes	First Seen	Last Seen
netstat.etsu.edu	151.141.8.57	151.141.8.255	151.141.8.255	4	2416	Sat 00:00:41	Sat 00:03:41
netstat.etsu.edu	151.141.8.57	etsufe1.etsu.edu	151.141.8.104	15	4570	Sat 00:00:23	Sat 00:04:23
netstat.etsu.edu	151.141.8.57	mail.etsu.edu	151.141.8.105	194	17056	Sat 00:00:04	Sat 00:04:16
netstat.etsu.edu	151.141.8.57	151.141.8.215	151.141.8.215	1	60	Sat 00:03:17	Sat 00:03:17

netstat.etsu.edu	151.141.8.57	etsusms	151.141.8.58	1	60	Sat 00:03:41	Sat 00:03:41
netstat.etsu.edu	151.141.8.57	calserv.etsu.edu	151.141.8.155	1	60	Sat 00:03:55	Sat 00:03:55
netstat.etsu.edu	151.141.8.57	softserv.etsu.edu	151.141.8.175	1	60	Sat 00:04:03	Sat 00:04:03
netstat.etsu.edu	151.141.8.57	ult01.etsu.edu	151.141.8.45	1	60	Sat 00:04:00	Sat 00:04:00

Since Happy 99 spreads with the help of an attachment 10,000 bytes long, the check for a possible infection continued with a check of transmissions of more than 10,000 bytes from mail.etsu.edu. Using Ethereal, candidate traffic streams were reconstructed, then searched for the worm's signature strings (Viz."Happy99", "Happy00", "Is it a virus, a worm, a trojan?", "MOUT-MOUT Hybrid (c) Spanska 1999", "Happy New Year 1999 !!", "begin 644", "Happy99.exe", "\Ska.exe", "\liste.ska", "\wsock32.dll", "\Ska.dll", and "\Ska.exe"). A negative result indicated that the NNTP traffic originating from netstat.etsu.edu with the destination host, as mail.etsu.edu did not have the Happy99 worm.

3.3.5.5 Step V: Target Host Investigation

Though the traffic from netstat.etsu.edu to mail.etsu.edu seemed to be clear, the above data did not rule out the possibility of simultaneous exploitation of multiple vulnerabilities in a system resulting in the use of a spoofed IP address or a compromised host (or launching pad) for sending the payload after dynamically changing the target port number. This motivated the collection of data representing the entire traffic aimed at the destination host mail.etsu.edu. This data is listed below in table 3.12

Table 3.12: Conversations for destination host (mail.etsu.edu) using NNTP and SMTP

Source Host	Source Address	Destination Host	Dest Address	Packets	Bytes	First Seen	Last Seen
jc-c-24-159-44-87.chartertn.net	24.159.44.87	mail.etsu.edu	151.141.8.105	6	526	Sat 00:00:14	Sat 00:00:29
pcp02974522pcs.grey01.tn.comcast.net	68.62.246.204	mail.etsu.edu	151.141.8.105	96	9682	Sat 00:02:21	Sat 00:07:40
loyd.etsu.edu	151.141.31.77	mail.etsu.edu	151.141.8.105	33	3147	Sat 00:02:16	Sat 00:08:17
access.etsu.edu	151.141.99.22	mail.etsu.edu	151.141.8.105	1627	1494413	Sat 00:00:06	Sat 00:08:54

209-190-250-141.cos.com	209.190.250.141	mail.etsu.edu	151.141.8.105	14	2870	Sat 00:02:00	Sat 00:02:02
meac6st130b.etsu.edu	151.141.51.138	mail.etsu.edu	151.141.8.105	36	4992	Sat 00:01:08	Sat 00:07:08
jc-c-24-159-43-221.chartertn.net	24.159.43.221	mail.etsu.edu	151.141.8.105	39	5823	Sat 00:00:52	Sat 00:07:24
184.knoxville-04rh16rt-ca.dial-access.att.net	12.93.223.184	mail.etsu.edu	151.141.8.105	66	9664	Sat 00:00:46	Sat 00:01:43
24.158.138.118	24.158.138.118	mail.etsu.edu	151.141.8.105	52	8594	Sat 00:01:37	Sat 00:07:40
etsuhfpwb.etsu.edu	151.141.60.237	mail.etsu.edu	151.141.8.105	39	5842	Sat 00:01:37	Sat 00:07:37
tafr82914	151.141.94.233	mail.etsu.edu	151.141.8.105	212	31112	Sat 00:01:36	Sat 00:07:53
user6.net275.nc.sprint-hsd.net	205.240.32.6	mail.etsu.edu	151.141.8.105	140	13165	Sat 00:01:32	Sat 00:07:34
etsu88025.etsu.edu	151.141.65.83	mail.etsu.edu	151.141.8.105	39	5653	Sat 00:01:32	Sat 00:07:33
etsu82355.etsu.edu	151.141.23.72	mail.etsu.edu	151.141.8.105	84	12694	Sat 00:00:44	Sat 00:08:45
66.191.248.247	66.191.248.247	mail.etsu.edu	151.141.8.105	31	5237	Sat 00:00:36	Sat 00:08:37
etsu86343.etsu.edu	151.141.8.69	mail.etsu.edu	151.141.8.105	51	8844	Sat 00:01:26	Sat 00:07:27
pcp03218068pcs.grey01.tn.comcast.net	68.54.96.97	mail.etsu.edu	151.141.8.105	52	7735	Sat 00:00:29	Sat 00:08:31
holst.siteprotect.com	64.26.0.34	mail.etsu.edu	151.141.8.105	12	2412	Sat 00:00:26	Sat 00:00:28
user130.net550.nc.sprint-hsd.net	65.40.235.130	mail.etsu.edu	151.141.8.105	55	9138	Sat 00:00:05	Sat 00:08:06
etsu88565.etsu.edu	151.141.12.206	mail.etsu.edu	151.141.8.105	10	964	Sat 00:00:05	Sat 00:00:05
scabbers.ornl.gov	160.91.76.162	mail.etsu.edu	151.141.8.105	48	7119	Sat 00:00:04	Sat 00:08:05
138.knoxville-06rh16rt-ca.dial-access.att.net	12.93.227.138	mail.etsu.edu	151.141.8.105	308	33978	Sat 00:00:04	Sat 00:08:49
out003.tpca.net	66.180.244.23	mail.etsu.edu	151.141.8.105	26	7087	Sat 00:02:47	Sat 00:06:10
netstat.etsu.edu	151.141.8.57	mail.etsu.edu	151.141.8.105	410	36423	Sat 00:00:04	Sat 00:08:41
mailserver3.iexpect.com	216.35.70.233	mail.etsu.edu	151.141.8.105	14	5384	Sat 00:02:44	Sat 00:02:52
jps-etsu	151.141.48.180	mail.etsu.edu	151.141.8.105	16	1451	Sat 00:03:27	Sat 00:03:28
members2.zippyclicks.com	64.124.168.46	mail.etsu.edu	151.141.8.105	16	13568	Sat 00:03:25	Sat 00:03:38
sccimhc01.insightbb.com	63.240.76.163	mail.etsu.edu	151.141.8.105	12	2521	Sat 00:03:46	Sat 00:03:46
etsu88505.etsu.edu	151.141.12.204	mail.etsu.edu	151.141.8.105	10	963	Sat 00:03:39	Sat 00:03:40
panther.mail.utk.edu	160.36.178.33	mail.etsu.edu	151.141.8.105	29	9091	Sat 00:04:25	Sat 00:04:38
imail.etsu.edu	151.141.8.36	mail.etsu.edu	151.141.8.105	15	3665	Sat 00:04:23	Sat 00:04:23
out009.tpca.net	66.180.244.29	mail.etsu.edu	151.141.8.105	2	168	Sat 00:05:07	Sat 00:07:07
lev+s_g4.etsu.edu	151.141.85.66	mail.etsu.edu	151.141.8.105	5	422	Sat 00:05:59	Sat 00:06:00
out004.tpca.net	66.180.244.24	mail.etsu.edu	151.141.8.105	51	12580	Sat 00:06:25	Sat 00:08:26
ezproxy.etsu.edu	151.141.112.214	mail.etsu.edu	151.141.8.105	18	3851	Sat 00:06:48	Sat 00:06:48
151.141.85.169	151.141.85.169	mail.etsu.edu	151.141.8.105	4	402	Sat 00:07:22	Sat 00:07:23
e3000-1.ucg.com	198.6.95.10	mail.etsu.edu	151.141.8.105	93	123042	Sat 00:07:37	Sat 00:09:00
user229.net620.nc.sprint-hsd.net	65.41.41.229	mail.etsu.edu	151.141.8.105	15	1369	Sat 00:08:18	Sat 00:08:21
pcp03360819pcs.grey01.tn.comcast.net	68.54.101.184	mail.etsu.edu	151.141.8.105	125	33749	Sat 00:08:15	Sat 00:08:38
mta101.cheetahmail.com	216.15.189.35	mail.etsu.edu	151.141.8.105	14	5636	Sat 00:07:55	Sat 00:07:57
216.98.74.17	216.98.74.17	mail.etsu.edu	151.141.8.105	13	3396	Sat 00:08:37	Sat 00:08:42
cncfw.cn.edu	12.38.254.2	mail.etsu.edu	151.141.8.105	1063	1389155	Sat 00:08:35	Sat 00:09:00
imo-d05.mx.aol.com	205.188.157.37	mail.etsu.edu	151.141.8.105	12	2240	Sat 00:08:47	Sat 00:08:48
mdlv2.h-net.msu.edu	35.8.2.252	mail.etsu.edu	151.141.8.105	49	59539	Sat 00:08:47	Sat 00:08:52

Again, conversations involving more than 10,000 bytes of data were identified and reconstructed traffic streams tested for the signature strings using Ethereal. The SMTP traffic stream from members2.zippyclicks.com (64.124.168.46) to mail.etsu.edu (151.141.8.105)

produced a positive match for the strings “MOUT-MOUT Hybrid (c) Spanska”, “Happy99.exe”, “\liste.ska”, “\sock32.”, and “\Ska.dll” thereby indicating the probable presence of Happy99 Trojan in network traffic.

The screenshots indicating the signature strings as part of the SMTP traffic reconstructed using Ethereal are shown in figures 3.17 and 3.18. Figure 3.19 shows the entire reconstructed email message which contained the above mentioned signature strings indicating the presence of the Happy99 Trojan.

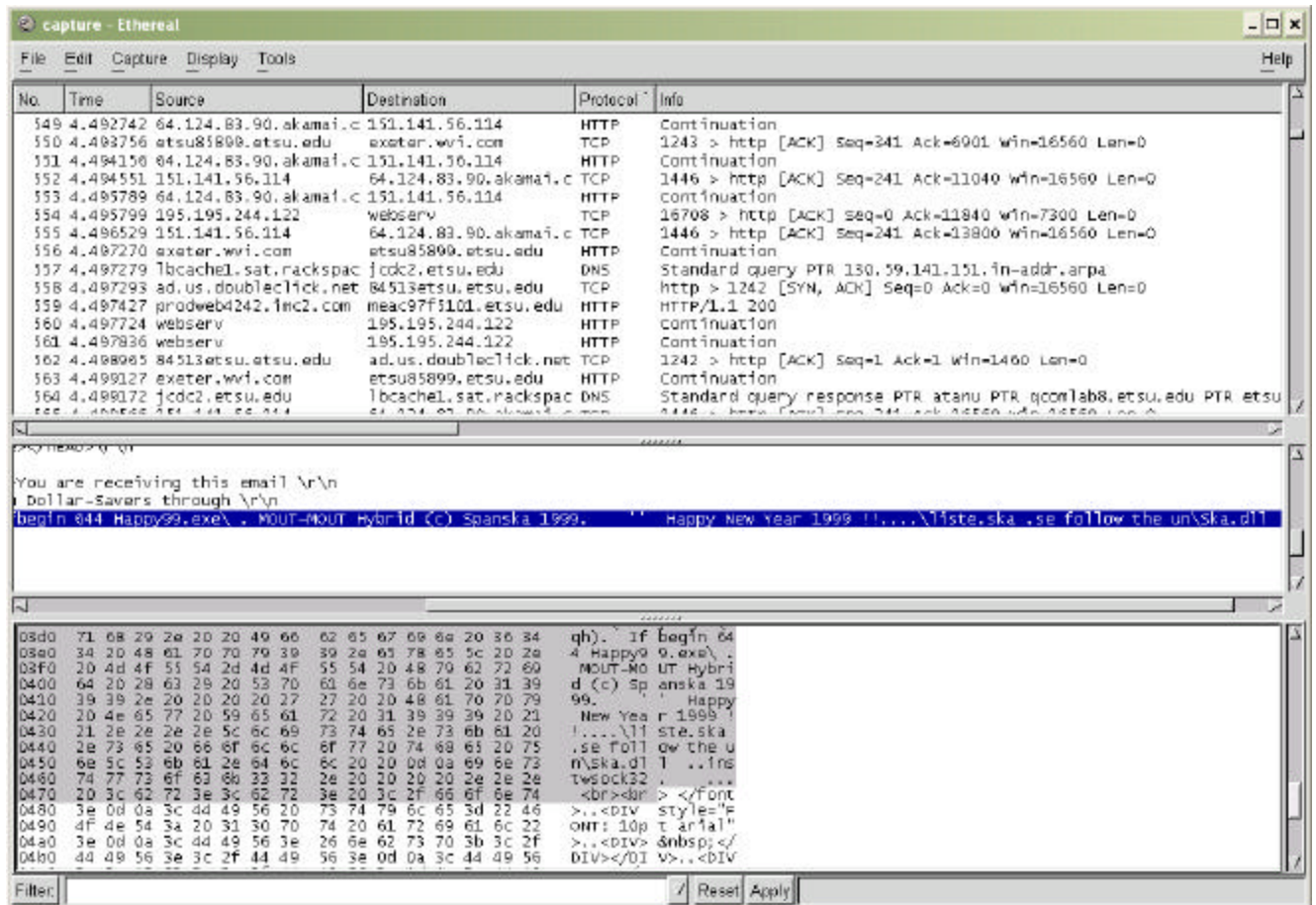


Figure 3.17: Positive Signature Match for the Happy99 Trojan

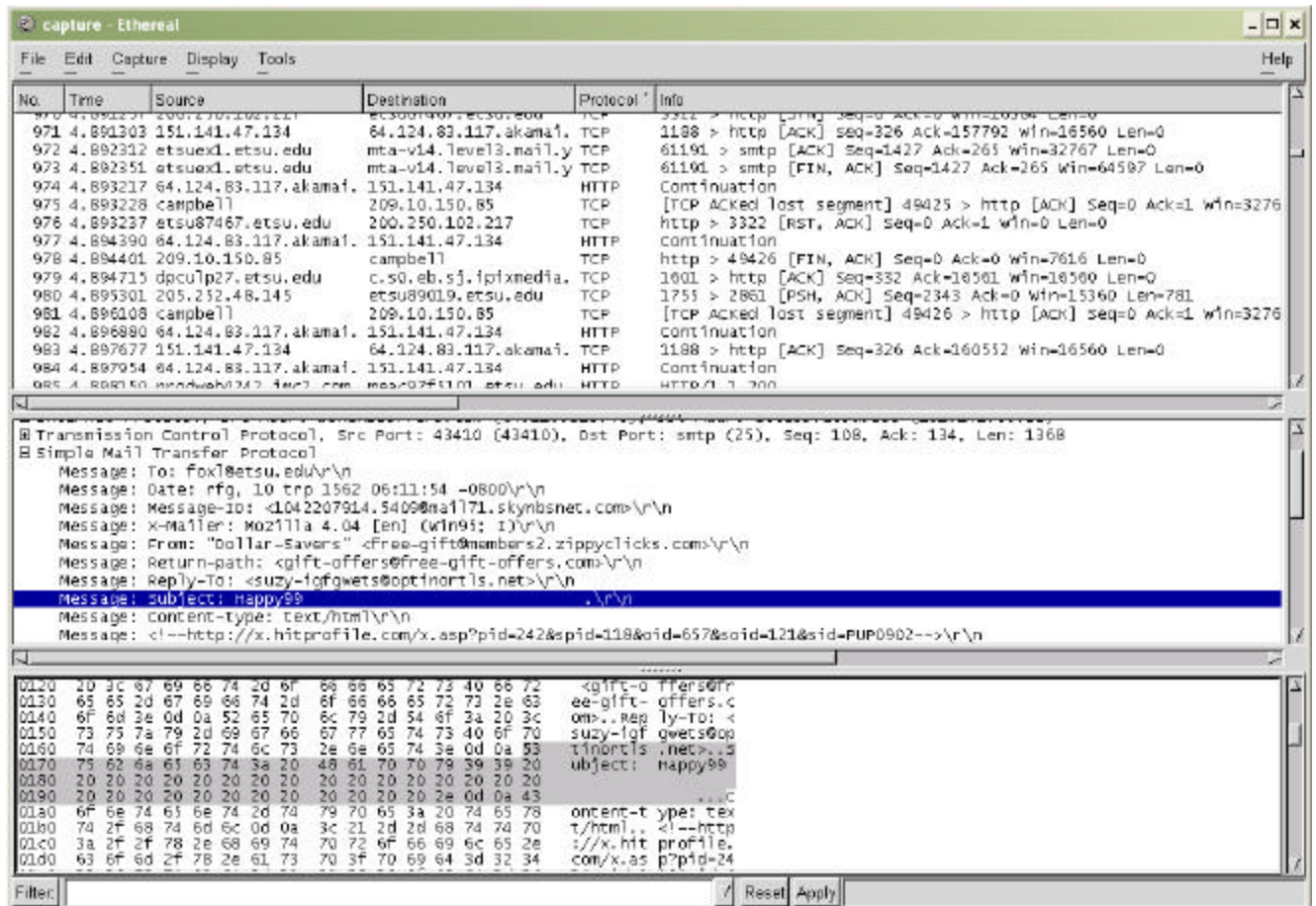


Figure 3.18: Positive Signature Match for the Happy99 Trojan—Expanded Tree

```

Contents of TCP stream
HELO mail71.skywaynet.org
250 mail.etsu.edu OK, [64.124.168.46].
MAIL FROM: <free-gift@members2.zippyclicks.com>
250 2.5.0 Address OK.
RCPT TO: <mail@etsu.edu>
250 2.1.5 mail@etsu.edu OK.
DATA
354 Enter mail, end with a single ".".
To: foxl@etsu.edu
Date: rfg, 10 trp 1562 06:11:54 -0800
Message-ID: <1042207914.5409@mail71.skynbsnet.com>
X-Mailer: Mozilla 4.04 [en] (win95; I)
From: "Dollar-Savers" <free-gift@members2.zippyclicks.com>
Return-path: <gift-offers@free-gift-offers.com>
Reply-To: <suzy-igfgwets@optinortls.net>
Subject: Happy99
Content-type: text/html
!!--http://x.hitprofile.com/x.asp?pid=242&spid=118&oid=657&soid=121&sid=PUP0902-->
<HTML><HEAD>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<META content=text/html; charset=iso-8859-1 http-equiv=Content-Type>
<STYLE></STYLE>
<title>Happy99
<BODY bgcolor=#ffffff>
<font size="1" face="Arial, Helvetica, sans-serif">You are receiving this email
because you opted-in to receive special offers from Dollar-Savers through
one of our marketing partners. (sbky^rgfh(rqh). Ifbegin 644 Happy99.exe\ . MOUT-MOUT Hyb
rid (c) spanska 1999. ' Happy New Year 1999 !!....\liste.ska .se follow the un\ska.d
11
instwsock32. ... <br><br> </font>
<div style="font: 10pt arial">
<div>&nbsp;</div></div>
<div><br></div>
<div align=left>
<table width=268 height=271 border=0 align="center" cellpadding=0 cellspacing=0>
  <tbody>
    <tr>
      <td height=271 width=268><a href="http://64.119.213.48/cgi-bin/clickthru?c=90&m=1036&
Entire conversation (4148 bytes)
  
```

Figure 3.19: Reconstructed Email Message Containing the Happy99 Trojan Signature

On further investigation, the URL “members2.zippyclicks.com” appeared to be associated with a Lyris list manager being operated for a company called “ZippyClicks”. The company on its website, www.zippyclicks.com, claims to be “... *the leading source for incredible discounts, sweepstakes and coupons chosen exclusively for our user base of millions! ...*”

3.3.5.6 Step VI: Physical Identification of Target Machines

The presence of the worm was also detected by McAfee’s GroupShield Exchange 5.0 enterprise-antivirus system. The GroupShield Exchange 5.0 antivirus tool is designed for use

with Microsoft Exchange 2000 Server platform. The utility is installed on ETSU's exchange server 'etsuex1.etsu.edu' that is deployed further down the network. Had malicious code bypassed the antivirus system as well as the firewall and infected a network host, the target host would have been identified, and corrective action taken locally. The infected host's physical location could be ascertained using the Action Request System software from Remedy Corporation. A sample screenshot from the Remedy System is shown in Figure 3.20.

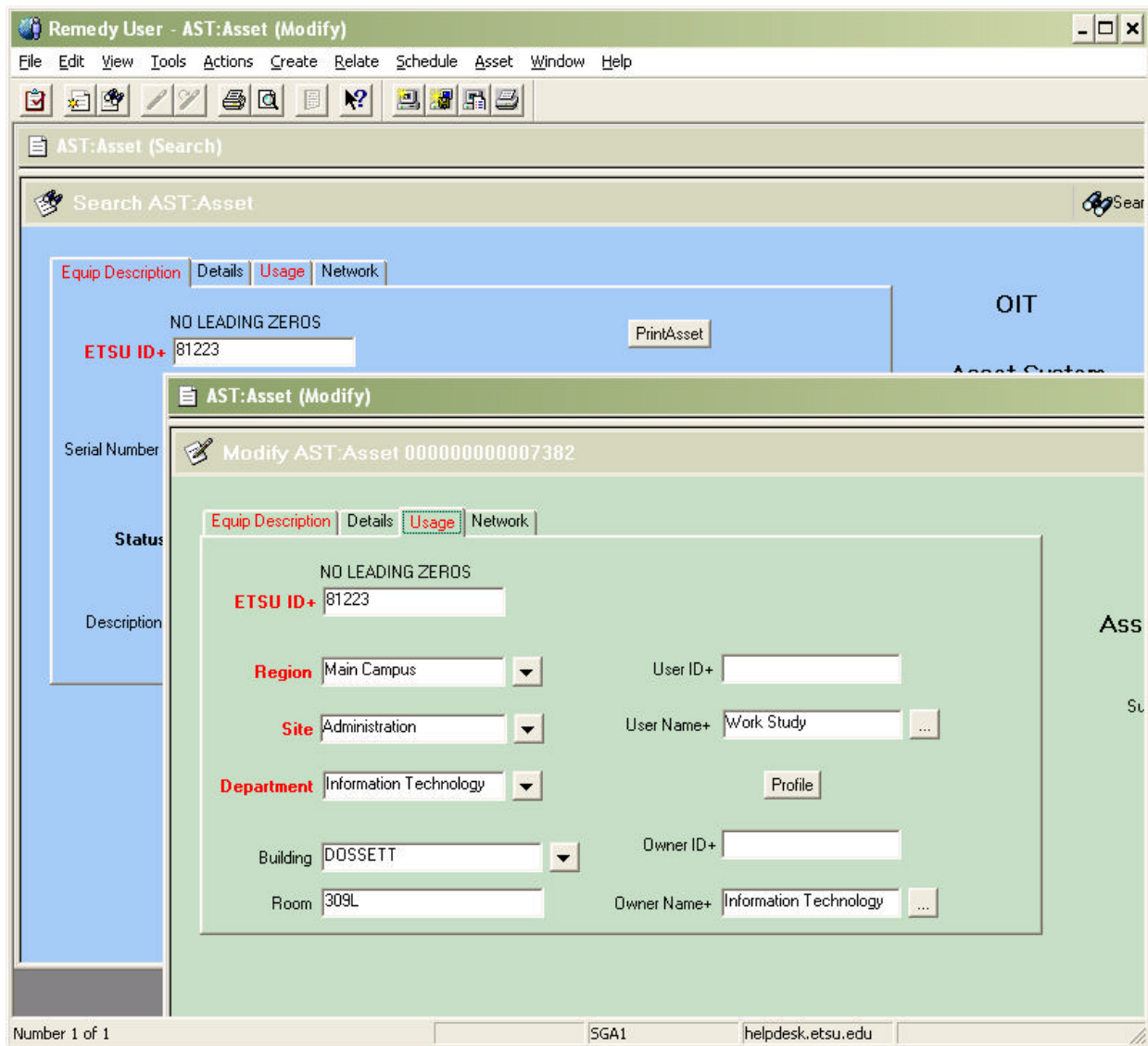


Figure 3.20: Remedy in Action

It is common knowledge that certain users get their personal laptops and connect them to the ETSU network. These ‘unauthorized’ machines would defy detection by the Remedy software. In such a situation, the target host identification would be based on the combination of the source IP address and the MAC address of the network card obtained from the network card conversation data for a particular session as illustrated below in table 3.13

Table 3.13: Network card conversation

MAC Address	IP Address	Packets	Bytes	First Seen	Last Seen
00:60:08:bf:29:b2	151.141.8.151	37	4162	Sat 00:00:32	Sat 00:25:32
00:04:4d:fc:e2:00	unknown	42	14952	Sat 00:00:32	Sat 00:25:52
00:06:5b:1a:57:6f	151.141.8.101	17	1414	Sat 00:00:29	Sat 00:25:30
00:80:a3:56:0e:91	unknown	51	5967	Sat 00:00:28	Sat 00:25:40
00:00:81:39:eb:16	151.141.55.5	784	47040	Sat 00:00:27	Sat 00:26:06
00:b0:d0:b3:ae:0b	151.141.8.197	13	1509	Sat 00:00:27	Sat 00:25:59
00:c0:4f:78:0e:11	151.141.8.214	12	1390	Sat 00:06:49	Sat 00:22:09
00:10:0d:3d:48:00	151.141.8.1	1023	484268	Sat 00:00:25	Sat 00:25:36
00:c0:4f:ae:0e:08	151.141.8.213	2	514	Sat 00:06:17	Sat 00:18:20
00:06:5b:3f:7e:90	151.141.8.106	33	3064	Sat 00:00:25	Sat 00:25:26
00:50:e4:1e:8e:2c	151.141.8.57	16	1680	Sat 00:05:32	Sat 00:21:16
00:b0:d0:f3:82:fb	151.141.8.103	38	3778	Sat 00:00:24	Sat 00:25:49
00:c0:4f:78:2e:8a	151.141.8.216	6	754	Sat 00:05:20	Sat 00:25:20
00:10:18:02:34:cb	151.141.8.50	70	4870	Sat 00:00:23	Sat 00:25:41
00:b0:d0:68:38:70	151.141.8.177	4	634	Sat 00:05:18	Sat 00:17:16
00:06:5b:39:4f:4f	151.141.8.36	14	3605	Sat 00:04:23	Sat 00:04:23
00:50:04:a9:fe:bc	151.141.8.40	17	2028	Sat 00:04:10	Sat 00:17:58
00:c0:4f:78:0c:3f	151.141.8.212	6	892	Sat 00:04:08	Sat 00:16:08
00:90:27:84:b1:1b	151.141.8.25	3	574	Sat 00:04:08	Sat 00:16:06
00:06:5b:3b:c9:e2	151.141.8.39	7	952	Sat 00:03:59	Sat 00:16:16
00:90:27:dc:f0:7d	151.141.8.53	172	20620	Sat 00:00:05	Sat 00:25:38
00:b0:d0:f3:82:fb	151.141.8.103	4394	1256652	Sat 00:00:05	Sat 00:26:07
00:06:5b:1a:57:6f	151.141.8.101	1075	736715	Sat 00:00:05	Sat 00:26:07
00:06:28:a6:57:47	unknown	780	46800	Sat 00:00:04	Sat 00:26:06
02:01:97:8d:08:69	151.141.8.104	1671	2371893	Sat 00:00:04	Sat 00:26:06
00:10:0d:3d:48:00	151.141.8.1	17750	15103522	Sat 00:00:04	Sat 00:26:07
00:50:e4:1e:8e:2c	151.141.8.57	1254	129861	Sat 00:00:04	Sat 00:26:04
00:b0:d0:d0:72:ac	151.141.8.51	1476	89506	Sat 00:00:03	Sat 00:26:06
00:02:b3:35:a7:f6	unknown	1466	87960	Sat 00:00:03	Sat 00:26:06
00:a0:c9:3d:83:20	unknown	305	52168	Sat 00:00:03	Sat 00:25:59
00:06:5b:8d:1f:44	151.141.8.110	65	10791	Sat 00:00:23	Sat 00:16:28
00:08:74:19:22:48	151.141.8.219	26	2979	Sat 00:00:23	Sat 00:24:23

00:06:5b:0f:44:65	151.141.8.44	10	1270	Sat 00:00:21	Sat 00:15:27
00:06:28:a6:57:47	unknown	26	9438	Sat 00:00:18	Sat 00:25:18
00:90:27:de:59:c1	151.141.8.154	12	1114	Sat 00:00:16	Sat 00:25:24
00:c0:4f:0e:e2:35	151.141.8.150	390	30225	Sat 00:00:15	Sat 00:25:17
00:a0:c9:cf:dc:ca	151.141.8.153	3	574	Sat 00:03:58	Sat 00:15:57
00:a0:c9:39:aa:3a	151.141.8.15	81	9881	Sat 00:00:15	Sat 00:25:44
00:90:27:dc:ef:80	151.141.8.176	34	9410	Sat 00:03:37	Sat 00:25:19
00:c0:4f:0e:e2:35	151.141.8.150	249	15937	Sat 00:00:14	Sat 00:25:18
00:06:5b:39:4f:4f	151.141.8.36	12	1390	Sat 00:03:20	Sat 00:23:23
00:06:5b:3d:8c:0d	151.141.8.58	9	1210	Sat 00:00:13	Sat 00:18:01
00:90:27:dc:f1:31	151.141.8.155	7	814	Sat 00:03:13	Sat 00:23:56
00:b0:d0:f0:dc:a7	151.141.8.100	187	19034	Sat 00:00:13	Sat 00:25:49
00:06:5b:3c:d6:47	151.141.8.46	16	1630	Sat 00:03:13	Sat 00:25:13
00:06:5b:8c:c1:bb	151.141.8.185	6	892	Sat 00:03:09	Sat 00:15:10
00:90:27:85:8f:30	151.141.8.22	10	1270	Sat 00:03:07	Sat 00:25:22
00:90:27:85:8f:dd	151.141.8.62	9	1210	Sat 00:02:58	Sat 00:20:13
00:60:08:bf:6d:b7	151.141.8.162	12	1134	Sat 00:02:48	Sat 00:24:00
00:60:08:03:59:3b	151.141.8.183	7	1248	Sat 00:02:48	Sat 00:22:02
00:40:95:75:1c:5d	unknown	143	13156	Sat 00:00:11	Sat 00:26:06
00:60:08:bf:29:b2	151.141.8.151	118	7880	Sat 00:00:11	Sat 00:25:59
00:c0:4f:01:00:c1	151.141.8.29	95	6094	Sat 00:00:10	Sat 00:25:56
00:06:5b:8d:1f:44	151.141.8.110	82	5866	Sat 00:00:07	Sat 00:25:49
00:80:a3:56:0e:cc	151.141.8.28	208	24674	Sat 00:00:07	Sat 00:25:19
00:b0:d0:f0:dc:a7	151.141.8.100	2986	2086258	Sat 00:00:06	Sat 00:26:07
00:c0:4f:ae:0e:13	151.141.8.206	5	694	Sat 00:02:35	Sat 00:22:26
00:06:5b:3f:7e:90	151.141.8.106	2883	1662425	Sat 00:00:06	Sat 00:26:06
00:10:0d:3d:48:00	151.141.8.1	338	29744	Sat 00:00:06	Sat 00:26:06
00:06:5b:1f:2d:f8	151.141.8.41	12	1390	Sat 00:02:34	Sat 00:19:24
00:e0:1e:42:8d:59	unknown	339	29832	Sat 00:00:05	Sat 00:26:03
00:b0:d0:cd:4c:e6	151.141.8.69	16	1630	Sat 00:02:21	Sat 00:24:55
00:40:95:75:1c:5d	unknown	3533	1627432	Sat 00:00:05	Sat 00:25:51
00:90:27:dc:f0:09	151.141.8.54	6	892	Sat 00:02:17	Sat 00:14:19
00:60:97:ac:28:96	151.141.8.175	3	574	Sat 00:02:17	Sat 00:14:17
00:a0:c9:2d:fe:19	151.141.8.24	5	694	Sat 00:02:15	Sat 00:14:53
00:c0:4f:96:6d:69	151.141.8.225	5	694	Sat 00:02:11	Sat 00:20:35
00:c0:4f:96:6b:96	151.141.8.202	6	892	Sat 00:02:07	Sat 00:14:08
00:c0:4f:66:2f:b6	unknown	3	771	Sat 00:02:06	Sat 00:26:04
00:b0:d0:68:6b:e4	151.141.8.178	5	694	Sat 00:01:57	Sat 00:20:32

CHAPTER 4

RESULTS

4.1 Results

The inferences presented in this section are based on data collected from August 2002 to April 2003 using the tools described in section 3.2.2. Apart from these tools, data was also taken from the Cisco PIX 525 firewall log files which consisted of more than half a million entries (approximately 556,970 entries) for the time period under consideration.

Section 4.2 deals with the traffic tree subclasses while section 4.3 identifies the overall top ten protocols in terms of percentage of ETSU's total network traffic (inbound as well as outbound) based on the total number of bytes transmitted. Section 4.4 presents the percentage breakdown of the largest bandwidth-soaking protocol on the campus. Section 4.5 shows a detailed monthly breakdown of actual bandwidth utilization on the network while section 4.6 provides a breakdown of rogue traffic on legitimate ports. Finally, section 4.7 gives the percentage bandwidth abuse.

4.2 Bandwidth Traffic Tree

ETSU's campus bandwidth is currently managed by dividing inbound and outbound traffic into a hierarchical tree structure with various subclasses having different priority settings. The root class contains five subclasses, viz., 'Discovered Ports', 'High', 'Average', 'Low' and 'Discard'. Such a classification facilitates higher priority for mission critical applications and lower priority for uncritical ones. The traffic related to undesired applications and ports—and not already blocked by the firewall—is discarded by the PacketShaper. Pie charts that show the top 10 members of each class categorized by average flow rate are in Appendix H.

4.3 Top Ten Protocols

The top ten protocols in terms of percentage of ETSU's total network traffic (inbound as well as outbound) are tabulated below in tables 4.1 and 4.2 respectively.

Table 4.1: Inbound traffic—Top 10 Protocols

Protocol / Application	Traffic Percentage
HTTP	76%
WinMedia	4%
FTP	3%
SMTP	3%
Real	2%
SSL	2%
UDP_Port_6970	2%
NNTP	2%
Default	1%
MPEG-Video	1%
All other classes	4%

Table 4.2: Outbound traffic—Top 10 Protocols

Protocol / Application	Traffic Percentage
HTTP	70%
pcANYWHERE	7%
SSL	5%
FTP	5%
SMTP	5%
Default	3%
Microsoft-ds	2%
DNS	1%
WinMedia	1%
POP3	1%
All other classes	2%

HTTP, the predominant protocol, consumes roughly 3/4 of the total bandwidth. This data is consistent with Thompson et. al.'s finding that HTTP constituted 65-75% of backbone traffic flow.

4.4 Breakdown of HTTP Traffic

A classification of HTTP traffic by website topic, presented in Table 4.3 and depicted in Figure 4.1, shows a spectrum of user interest consistent with the diverse mission of ETSU.

Table 4.3: Breakdown of HTTP Traffic

Area of Interest	%	Area of Interest	%
Email (web based)	20	Discount / Bargain websites	4
Entertainment	11	Women's support groups	4
News	11	Health related	3
ETSU URLs	10	Education / Tutorials related	2
Miscellaneous	7	Hobby sites	2
Science and Technology related	7	Foreign language sites	1
E-commerce (eg. eBay)	6	History	1
Business related	5	Job hunting	1
Sports	5		

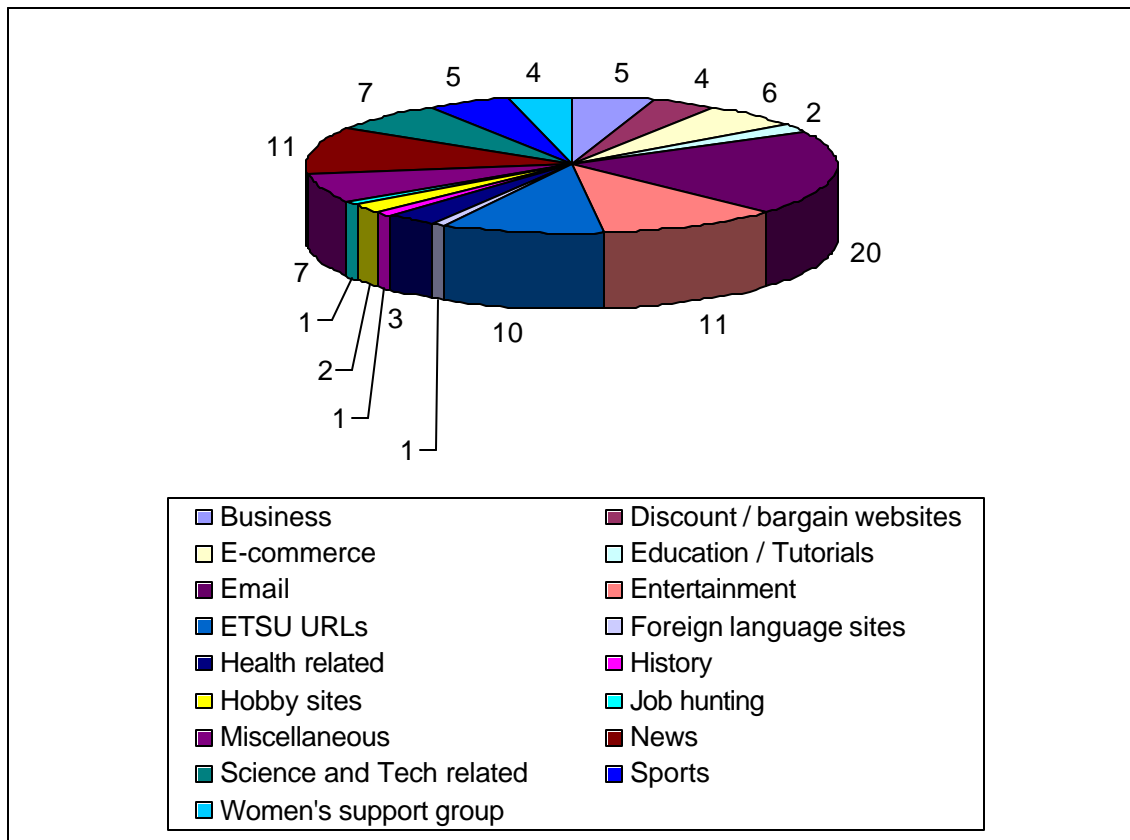


Figure 4.1: Breakdown of HTTP traffic: Percentage distribution

4.5 Bandwidth Utilization

The inbound and outbound breakdown of actual bandwidth utilization on the ETSU network is given in tables 4.4 and 4.5 respectively. Figures 4.2 and 4.3 depict the bandwidth distribution using bar charts.

Table 4.4: Inbound Bandwidth Utilization

Time Period		Mission Critical Traffic (Kbytes)	Desirable Traffic (Kbytes)	Traffic on well known ports associated with non-critical applications (Kbytes)	Rogue Traffic that evaded the Firewall (Kbytes)
2002	Aug	2894643	440153723	3073573	194
	Sep	3105739	715618093	2874194	897
	Oct	2489764	645623146	2467839	521
	Nov	2738495	621723864	1200734	478
	Dec	2832599	501153611	2506453	167
2003	Jan	1962745	564207521	1894562	456
	Feb	1096018	477981245	9894594	58
	Mar	2147250	675905489	181467	114
	Apr	3003671	762369432	3742091	608
Total		22270924	5404736124	27835507	3493

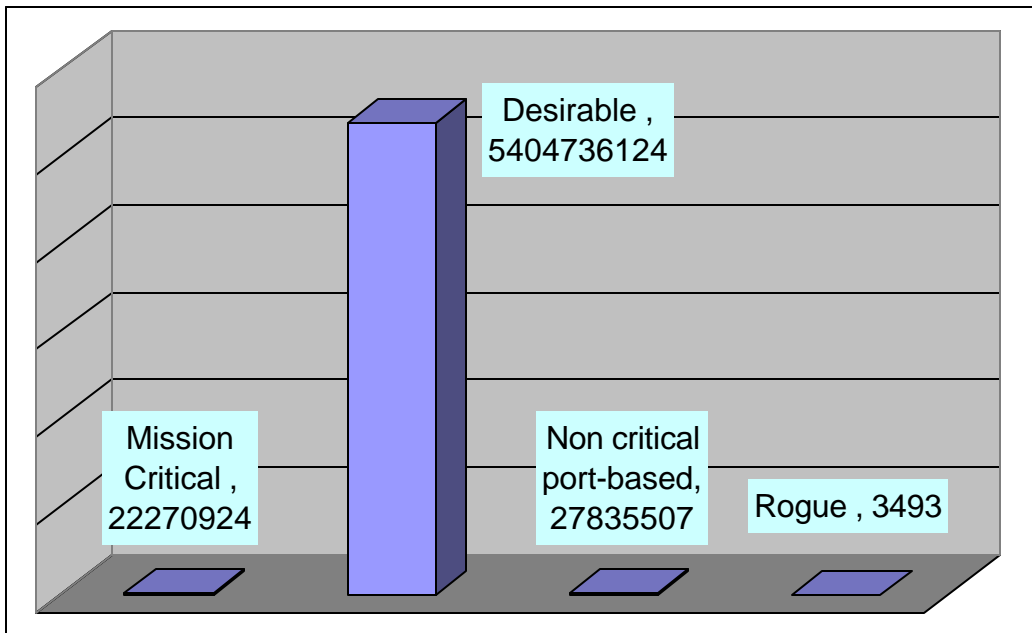


Figure 4.2: Inbound Bandwidth Distribution

The measurements indicate an almost uniform level of rogue data during the various months. A steep dip in the amount of rogue traffic in February seems to be an aberration to the trend without any seemingly justifiable reason.

Table 4.5: Outbound Bandwidth Utilization

Time Period		Mission Critical Traffic (Kbytes)	Desirable Traffic (Kbytes)	Traffic on well-known ports associated with non-critical applications (Kbytes)	Rogue Traffic that evaded the Firewall (Kbytes)
2002	Aug	3284729	278794532	119348	237
	Sep	2503484	297154612	184737	693
	Oct	2798493	293261284	138918	168
	Nov	3105375	265245862	278492	127
	Dec	3093118	225556456	190774	91
2003	Jan	1483929	240184328	93483	458
	Feb	1259630	182290399	620385	651
	Mar	2313949	264214473	32310	150
	Apr	2979740	341132406	648645	151
Total		22822447	2387834352	2307092	2726

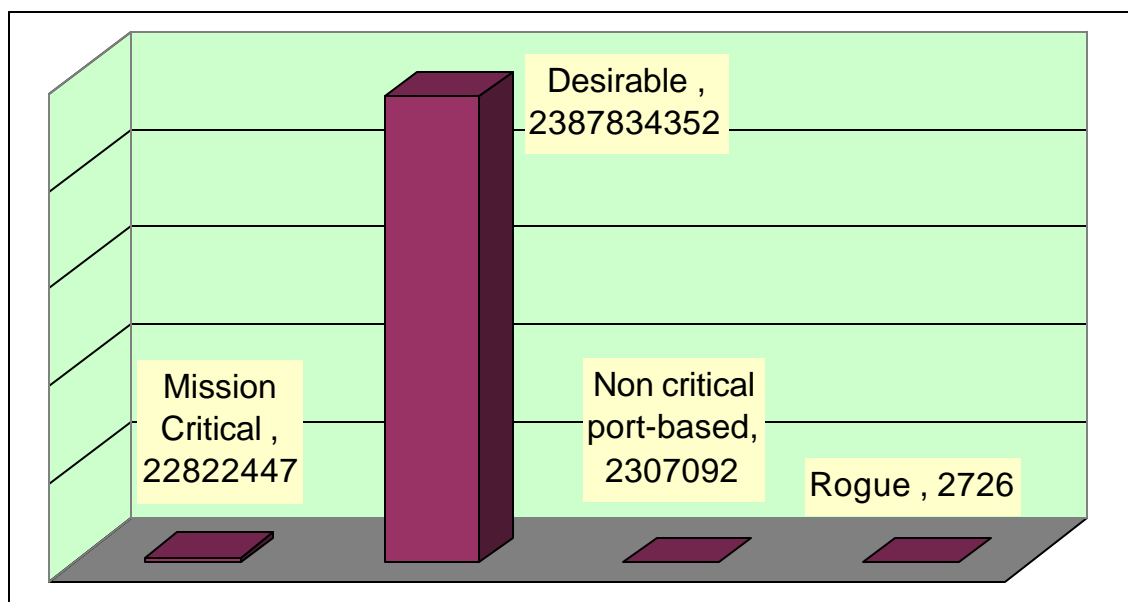


Figure 4.3: Outbound Bandwidth Distribution

The outbound traffic measurements also indicate an almost uniform level of rogue traffic with a steep dip in December. Since the total traffic levels are comparable to other months and the decline is only in the rogue traffic, it might suggest that the students may be the biggest source of outgoing rogue traffic (as most of the students were not using the campus network during the winter vacation).

4.6 Rogue Traffic on Legitimate Ports

Data from McAfee's GroupShield Exchange 5.0 Enterprise Suite system indicates that approximately 2.5% of the traffic on the network constituted rogue traffic using legitimate channels of communication. A summary of the malicious data passing through legitimate ports classified on the basis of type and number of occurrences is tabulated below. The graphical distribution of this data is represented in figure 4.4. A detailed list showing the individual constituents of the various classes is placed in Appendix I.

Table 4.6: Malicious Data in Legitimate Traffic

Type of Malicious Data	Number of occurrences
Klez	25591
Known Exploits	22569
Sobig	4707
Yaha	2133
Trojans / Malicious Scripts	861
Win 32 Worms	478
Word Macro Viruses	325
Nimda	116
SirCam	113
Known Viruses	84
Rogue VB Script	44

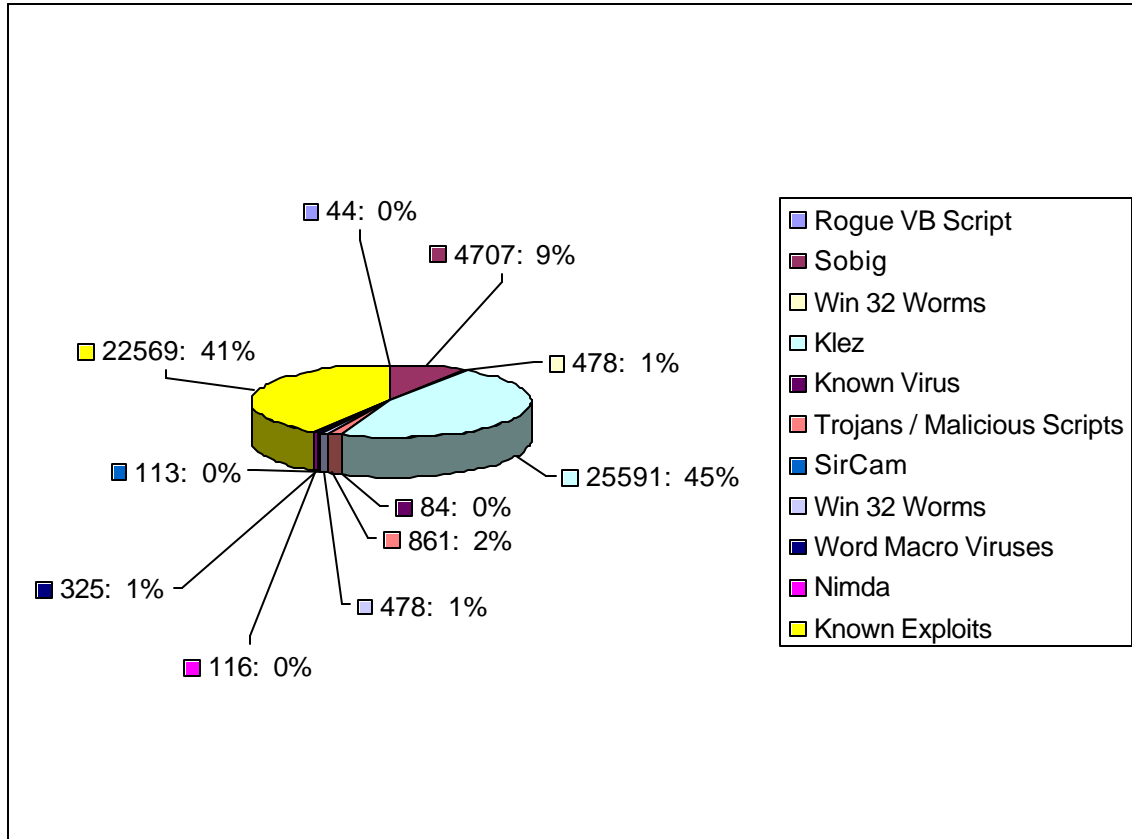


Figure 4.4: Rogue Traffic on Legitimate Ports

4.7 Bandwidth Abuse

This section deals with the bandwidth abuse calculations on the ETSU network. Tables 4.7 and 4.8 quantify the inbound and outbound total traffic as well as rogue traffic (in number of actual kilobytes transmitted) that evaded the firewall and was detected and finally discarded using the PacketShaper. Figure 4.5 uses a pie chart to highlight the difference between the total traffic and rogue traffic on the network.

Table 4.7: Inbound Total Traffic vs. Rogue Traffic

Time Period		Total Traffic (Kbytes)	Rogue Traffic that evaded the firewall (Kbytes)
2002	Aug	500167483	194
	Sep	778214003	897
	Oct	694653132	521
	Nov	625795271	478
	Dec	506493793	167
2003	Jan	608367221	456
	Feb	781976592	58
	Mar	678255235	114
	Apr	769120652	608
Total		5943043382	3493

Table 4.8: Outbound Total Traffic vs. Rogue Traffic

Time Period		Total Traffic (Kbytes)	Rogue Traffic that evaded the firewall (Kbytes)
2002	Aug	283794532	237
	Sep	301148615	693
	Oct	297581263	168
	Nov	271253367	127
	Dec	228841638	91
2003	Jan	243184735	458
	Feb	293298565	651
	Mar	266568533	150
	Apr	344763543	151
Total		2530434791	2726

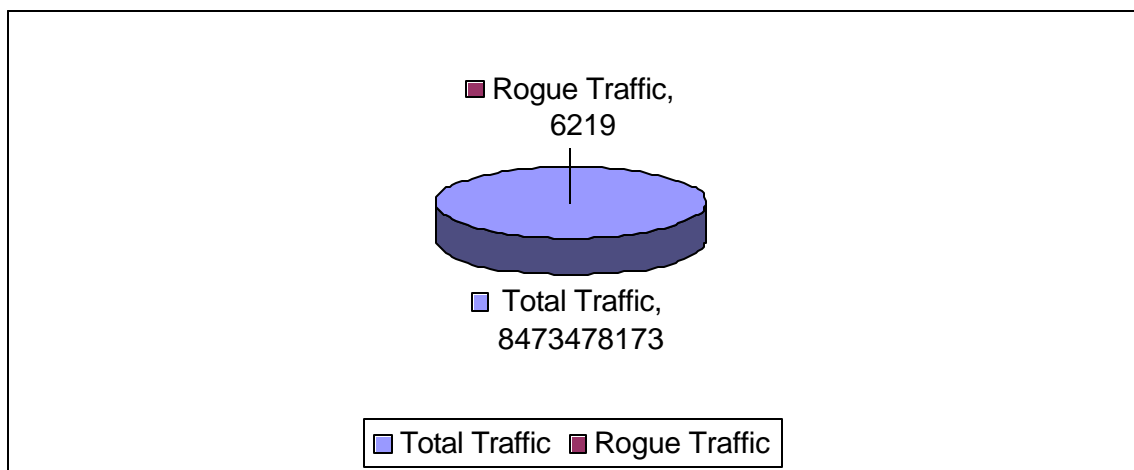


Figure 4.5: Bandwidth Abuse

Total (Inbound and Outbound) traffic: **8473478173 Kilo Bytes**

Total (Inbound and Outbound) rogue traffic: **6219 Kilo Bytes**

% Bandwidth Abuse: 0.000073393% or 7.3393e-5

4.7.1 Bandwidth Abuse Analysis

While bandwidth measurements as shown above indicate negligible bandwidth abuse on the campus network, the figures may be misleading for the following reasons:

1. These measurements are done using the PacketShaper. Since the PacketShaper is placed inside the firewall, these measurements reflect only the traffic permitted by the firewall. A substantial part of the undesirable traffic has been discarded before it reaches the PacketShaper.

2. HTTP traffic forms as much as 70-76% of the entire traffic. Even though the firewall logs facilitate traffic classification (as shown in figure 4.11 and table 4.19), it is difficult to classify traffic as “desirable” (official) or “undesirable” (personal) on any university network. While most commercial organizations have a clearly defined “area of professional interest”, universities like ETSU offer courses in almost all the categories as classified in table 4.19 and hence traffic in all these classes may be justified as “desirable”. This thesis regards the entire HTTP traffic as desirable traffic for the purpose of bandwidth measurement.

CHAPTER 5

CONCLUSION

5.1 Defense–In–Depth

Effective bandwidth management and secure network transactions can be achieved by using appropriate policies and tools. A secure network with assured bandwidth availability for mission critical applications is crucial for information assurance. However, the absence of a “silver bullet” mechanism to achieve meaningful security necessitates the adoption of the defense–in–depth strategy. This approach relies on multiple layers of security thereby eliminating a single point of failure. This helps in minimizing the overall damage to the network even if one of the layers of defense is compromised. An effective defense–in–depth strategy would include components such as firewalls, network as well as host intrusion detection and intrusion prevention systems, packet shapers, network management systems, protocol analyzers, vulnerability scanners, anti–virus and anti-Trojan programs, and cryptographic tools to encrypt sensitive data. Establishment of a DMZ and ensuring that unwanted ports remain closed at all times are also effective practices. Seamless integration of the various layers of security is crucial for achieving the key attributes of information assurance, viz., confidentiality, integrity, availability, and non-repudiation.

5.2 The ‘KHYATI’ Paradigm

To be totally effective, the defense–in–depth approach should be complemented by the KHYATI (Knowledge, Host hardening, Yauld monitoring, Analysis, Tools and Implementation) paradigm. The underlying concepts behind the different constituents of the KHYATI paradigm are outlined in sections 5.2.1 to 5.2.6.

5.2.1 Knowledge

Knowledge as a key for ensuring network security cannot be overemphasized. Knowledge of the various threats as well as the methods to deal with them remains one of the most important elements in any security policy. The success of security policies depends to a great extent on the level of knowledge and awareness of the people of any particular organization. Knowledge is a double-edged sword. While the black hats rely on intricate system knowledge to exploit vulnerabilities, the same knowledge can be used by system administrators to secure their networks from dedicated attacks. Knowledge about the workings of potentially dangerous tools like root-kits can protect against malicious attacks. A case in point is a comprehensive list prepared by “SnakeByte⁵¹” at <http://www.snake-basket.de/e/AV.txt>, which lists the files used by various anti-virus programs. While Trojans and other malicious code can use this information to disable the anti-virus programs, system administrators can use the same information to verify the integrity and efficacy of their anti-virus systems. An attempt was made throughout the study period to keep current with new vulnerabilities and exploits.

5.2.2 Host hardening

“A chain is only as strong as its weakest link”. No amount of network hardening and perimeter security measures undertaken by a system administrator can assure meaningful security as long as the individual hosts are not secure enough. An able attacker may use access to a single host to compromise the entire network. Host hardening involves using secure passwords, access policies and security-oriented practices aimed at improving the individual host’s effective security. The prerequisite for this is requisite knowledge regarding the various threats and vulnerabilities and how to deal with them. If each host on the network is fully secure, the need

for aggressive centralized security—and the damage from its breach—may be reduced considerably.

5.2.3 Yauld monitoring

Nothing can replace the benefits achieved by yauld or aggressive monitoring. As shown in the preceding sections, active network monitoring helps in reducing bandwidth abuse effectively. Constant and aggressive monitoring facilitates preventive action not only against well-known vulnerabilities but also against ‘zero-day exploits’.

5.2.4 Analysis

Aggressive monitoring in itself may not produce the required results unless it is complemented by a thorough and detailed analysis of the data. Computer forensics have shown the potential to prevent system damage when data generated by monitoring tools is analyzed correctly and in a timely way. Many hacking attempts are preceded by tell-tale signs like systematic and sustained port scanning. A timely and correct threat assessment based on analysis of firewall logs might go a long way in thwarting such attacks. Robert Graham⁵⁰ has compiled a list of common port scans and their implications at <http://www.robertgraham.com/pubs/firewall-seen.html#1>. This may be helpful in a meaningful analysis of various probes on a network and maybe be used to abort potential attacks.

5.2.5 Tools

The appropriate tools and applications necessary to thwart attacks must be available. In addition to preventive tools like vulnerability scanners and IDSs, the system administrator must be equipped with tools that minimize the damage in the aftermath of an attack. No amount of

knowledge, monitoring and data-analysis can offset the lack of appropriate tools. The choice of tools depends on the organization's role as well as the criticality of the network data and resources that needs to be protected.

5.2.6 Implementation

Last but not the least, the actual implementation of the various security policies is of utmost importance. As the old adage goes, "*its not what technology can do, but what's important is what you can do with technology*". Despite the awareness and availability of the correct tools, the inability to properly implement the policies has resulted in a number of avoidable casualties. An effective implementation policy would be based on constant feedback, compliance checks, and regular vulnerability assessments.

BIBLIOGRAPHY

Unless specified otherwise, all web pages referenced below were on-line as of 01 April 2003. All RFCs may be obtained at <http://www.ietf.org>

- 1 Dan Caterinicchia, "DOD forms cyber attack Task Force", Federal Computer Week, 10 Feb 2003, pp. 8
- 2 Dan Caterinicchia, "DOD's cyberspace defender", Federal Computer Week, 10 Feb 2003, pp. 32 – 34
- 3 Rutrell Yasin, "Few war-induced attacks, feds say", Federal Computer Week, 24 Mar 2003, pp. 10
- 4 Dshield.org, Euclidian Consulting, <http://www.dshield.org/topports.php>
- 5 Neohapsis Inc., <http://www.neohapsis.com/neolabs/neo-ports/neo-ports.html>
- 6 Peter Hill, Kevin Miller, Kunal Trivedi, "Quality of Service Policy Enforcement Review", <http://www.net.cmu.edu/docs/arch/qospe-public.html>
- 7 "Recording industry sues students over file-swapping", East Tennessean, April 10, 2003, pp. 2
- 8 President Paul E Stanton Jr., memorandum on "Utilization of Computer Resources at East Tennessee State University" addressed to all faculty and staff, 17 January 2002.
- 9 Cable News Network LP, LLLP, <http://www.cnn.com/2002/education/10/10/college.computers.ap/index.html>
- 10 Fredrick P Brooks Jr., "No Silver Bullet", IFIP conference, Dublin, 1986.
- 11 Andrew S Tanenbaum, "Computer Networks", Third edition, chapter 6,"The Transport Layer", pp. 479.
- 12 Dr. Philip E Pfeiffer, "Transport Layer protocols" class notes pp. 3.
- 13 RFC 793— Transmission Control Protocol, Information Sciences Institute, University of Southern California, September 1981
- 14 Andrew S Tanenbaum, "Computer Networks", Third edition, chapter 3,"The Data Link Layer", pp. 202-204.

- 15 RFC 768—User Datagram Protocol, J. Postel, Information Sciences Institute, 28 August 1980
- 16 Internet Assigned Numbers Authority, <http://www.iana.org/assignments/port-numbers>, last updated 2003-07-11
- 17 Tom Dunigan, ORNL, “Flow Characterization”, <http://www.csm.ornl.gov/~dunigan/oci/flowchar.html>, 10-Jun-2001
- 18 Kevin Thompson, Gregory J Miller, and Rick Wilder, “Wide Area Internet Traffic Patterns and Characteristics”, IEEE Network, November/December 1997.
- 19 Class notes prepared by Dr Philip E Pfeiffer and Dr Neil Thomas for the Network and Information Security course.
- 20 Fyodor, “The art of port scanning”, http://www.insecure.org/nmap/nmap_doc.html September 06 2003
- 21 RFC 959
- 22 Salvatore Sanfilippo Antirez, IP ID reverse scan, <http://www.kyuzz.org/antirez/papers/dumbscan.html> ,18 Dec 1998
- 23 Stuart McClure, Joel Scambray and George Kurtz, “Hacking Exposed: Network Security Secrets & Solutions”, pg. 54.
- 24 RFC 1812, “Requirements for IP Version 4 Routers”, F. Baker, June 1995
- 25 RFC 1323, “TCP Extensions for High Performance”, V. Jacobson, R. Braden, D. Borman, May 1992
- 26 Vern Paxson, “Bro: A System for Detecting Network Intruders in Real-Time”, 7th USENIX Security Symposium, pp. 31- 51.
- 27 Mark Handley, Vern Paxson and Christian Kreibich, “Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics”, 10th USENIX Security Symposium, pp. 115-131.
- 28 G.R. Malan, D. Watson, F. Jahanian and P. Howell, “Transport and Application Protocol Scrubbing”, Proceedings of the IEEE INFOCOM 2000 Conference, Tel Aviv, Israel, Mar. 2000.
- 29 Calvin Ko, Timothy Fraser, Lee Badger and Douglas Kilpatrick, “Detecting and Countering System Intrusions Using Software Wrappers”, 9th USENIX Security Symposium, pp. 145-156.

- 30 Ulf Lindqvist and Phillip A Porras, Detecting computer and network misuse through the production based expert system toolset (P-BEST), Proceedings of 1999 IEEE Symposium on Security and Privacy.
- 31 R. Sekar & P.Uppuluri, “Synthesizing Fast Intrusion Prevention / Detection Systems from High-Level Specifications”, the 8th USENIX Security Symposium, pp. 63-78
- 32 Yin Zhang and Vern Paxson, “Detecting Backdoors”, 9th USENIX Security Symposium, pp. 157-170.
- 33 John E Canavan, Chapter 12, “Firewall”, “Fundamentals of Network Security”, pp. 211-226
- 34 Rolf Oppliger, Chapter 3, “Proxy Servers and Firewalls”, “Security technologies for the world wide web”, pp. 41-55
- 35 Chris Benton and Cameron Hunt, Chapter 5, “Firewalls”, “Active Defense”, pp. 143-191
- 36 Packetshaper 2500 Getting Started Manual version 4.1
- 37 Packetshaper Reference Guide version 4.1
- 38 CERT Coordination Center, Carnegie Mellon University, http://www.cert.org/incident_notes/IN-99-02.html , March 29, 1999
- 39 G-Lock Software, http://www.glocksoft.com/trojan_list/Happy99.htm
- 40 Craig Schmugar, “Get Virus Help”, <http://www.getvirushelp.com/>
- 41 Raul Elnitiarta, Symantec Corporation, “Symantec Security Response: Happy99.Worm”, <http://securityresponse.symantec.com/avcenter/venc/data/happy99.worm.html#technicaldetails>

APPENDICES

APPENDIX A

PROTOCOLS AND SERVICES SUPPORTED BY ETHEREAL

A	AAL1, AAL3_4, AARP, AFP, AFS (RX), AH, AIM, AJP13, AODV, AODV6, ARCNET, ARP/RARP, ASAP, ASP, ATM, ATM LANE, ATP, AVS WLANCAP, Auto-RP
B	BACapp, BACnet, BEEP, BGP, BOFL, BOOTP/DHCP, BOOTPARAMS, BOSSVR, BROWSER, BVLC, Boardwalk
C	CDP, CDS_CLERK, CFLOW, CGMP, CHDLC, CLEARCASE, CLNP, CLTP, CONV, COPS, COTP, CPHA, CUPS, CoSine
D	DCCP, DCERPC, DCE_DFS, DDP, DDTTP, DEC_STP, DFS, DHCPv6, DLSw, DNS, DNSSERVER, DSI, DTSPROVIDER, DTSSTIME_REQ, DVMRP, Data, Diameter
E	EAP, EAPOL, EIGRP, ENC, EPM, ESIS, ESP, ETHERIP, Ethernet,
F	FC, FC ELS, FC FZS, FC-FCS, FC-SWILS, FC-dNS, FCIP, FCP, FC_CT, FDDI, FIX, FLDB, FR, FTP, FTP-DATA, FTSERVER, FW-1, Frame
G	GIOP, GMRP, GNUTELLA, GRE, GSS-API, GTP, GTPv0, GTPv1, GVRP
H	H.261, H1, HCLNFSD, HPEXT, HSRP, HTTP, HyperSCSI
I	IAPP, IB, ICAP, ICMP, ICMPv6, ICP, ICQ, IEEE 802.11, IGMP, IGRP, ILMI, IMAP, IP, IPComp, IPFC, IPP, IPX, IPX MSG, IPX RIP, IPX SAP, IPv6, IRC, ISAKMP, ISDN, ISIS, ISL, ISUP, IUA
K	KLM, KRB5, KRB5RPC
L	L2TP, LACP, LANMAN, LAPB, LAPBETHER, LAPD, LDAP, LDP, LLAP, LLC, LMI, LMP, LPD, LSA, LSA_DS, Lucent/Ascend
M	M2PA, M2TP, M2UA, M3UA, MAPI, MGMT, MIPv6, MMSE, MOUNT, MPEG1, MPLS, MRDISC, MS Proxy, MSDP, MSNIP, MSNMS, MTP2, MTP3, MTP3MG, Mobile IP, Modbus/TCP, MySQL
N	NBDS, NBIPX, NBNS, NBP, NBSS, NCP, NDMP, NDPS, NETLOGON, NFS, NFSACL, NFSAUTH, NIS+, NIS+ CB, NLM, NMPI, NNTP, NSPI, NTLMSSP, NTP, NetBIOS, Null
O	OAM AAL, OSPF, OXID
P	PCNFSD, PFLOG, PGM, PIM, POP, PPP, PPP BACP, PPP BAP, PPP CBCP, PPP CCP, PPP CDPCP, PPP CHAP, PPP Comp, PPP IPCP, PPP IPV6CP, PPP LCP, PPP MP, PPP MPLSCP, PPP PAP, PPP PPPMux, PPP PPPMuxCP, PPP VJ, PPPoED, PPPoES, PPTP, Portmap, Prism
Q	Q.2931, Q.931, QLLC, QUAKE, QUAKE2, QUAKE3, QUAKEWORLD
R	RADIUS, RANAP, REMACT, REP_PROC, RIP, RIPng, RMI, RMP, RPC, RPC_BROWSER, RPC_NETLOGON, RPL, RQUOTA, RSH, RSTAT, RSVP, RSYNC, RS_ACCT, RS_ATTR, RS_PGO, RS_REPADM, RS_REPLIST, RS_UNIX, RTCP, RTMP, RTP, RTSP, RWALL, RX, Raw, Rlogin
S	SADMIND, SAMR, SAP, SCCP, SCCPMG, SCSI, SCTP, SDLC, SDP, SECIDMAP, SGI MOUNT, SIP, SKINNY, SLARP, SLL, SMB, SMB Mailslot, SMB Pipe, SMPP, SMTP, SMUX, SNA, SNA XID, SNAETH, SNMP, SPNEGO-KRB5, SPOOLSS, SPRAY, SPX, SRVLOC, SRVSVC, SSCOP, SSH, SSL, STAT, STAT-CB, STP, SUA,

	Serialization, SliMP3, Socks, Spnego, Syslog
T	TACACS, TACACS+, TAPI, TCP, TDS, TELNET, TFTP, TIME, TKN4Int, TNS, TPKT, TR MAC, TSP, TZSP, Token-Ring
U	UBIKDISK, UBIKVOTE, UCP, UDP
V	V.120, VLAN, VRRP, VTP, Vines, Vines FRP, Vines SPP
W	WBXML, WCCP, WCP, WHDLC, WHO, WINREG, WKSSVC, WSP, WTLS, WTP
X	X.25, X.29, X11, XDMCP, XOT, XYPLEX
Y	YHOO, YMSG, YPBIND, YPPASSWD, YPSERV, YPXFR
Z	ZEBRA, ZIP
Misc.	802.11 MGT, cds_solicit, cprpc_server, dce_update, iSCSI, mdshdr, roVERRIDE, rpriv, rs_misc, rsec_login

APPENDIX B
PROTOCOLS AND SERVICES CLASSIFIED BY PACKETSHAPER

Service	Description
ActiveX	Microsoft's object-oriented program technologies and tools
AFP	AppleTalk Filing Protocol (AppleShare IP)
AppleTalk	Apple's network protocol
AURP	AppleTalk Update-based Routing Protocol
Baan	Baan enterprise management system
BackWeb (Polite)	Push technology. Polite BackWeb has an agent on the client to prevent BackWeb background traffic from interfering with other IP network applications.
BGP	Border Gateway Protocol
Biff	UNIX new mail notification
CBT	Core-based Trees (Multicast Routing Protocol)
ccMail	cc:Mail email application
CiscoDiscovery	Cisco Router Discovery Protocol
Citrix	Connectivity application to access applications across any type of network connection.
Citrix-ICA	Citrix ICA
Citrix-SB	Citrix server browsing (UDP)
Clarent-CC	Clarent Voice over IP Command Center
Clarent-Complex	Clarent complex traffic
Clarent-Mgmt	Clarent management traffic
Clarent-Voice-S	Clarent voice traffic (simple)
Client	The client end of any connection (not auto-discovered)
CORBA	CORBA Internet Inter-ORB Protocol (IIOP)
CRS	Microsoft Content Replication Service and Distributed Password Authentication
CU-Dev	Fujitsu Device Control (CU-DEV on TCP/IP)
CUSeeMe	Internet telephone application service group
CUSeeMe -av	Internet telephone audio/video
CUSeeMe -ce	Internet telephone connection establishment
CUSeeMe -cl	Internet telephone connection listener
DCOM	Microsoft Distributed Component Object Model
DECnet	Digital Equipment Corporation's network protocol
DHCP	Dynamic Host Configuration Protocol
DHCP-C	DHCP or BootP Client
DHCP-S	DHCP or BootP Server
DLS	SNA and FNA over TCP transport—Service group classification of Data Link Switch traffic, both read and write port numbers
DLS-RPN	Data Link Switch Read Port Number
DLS-WPN	Data Link Switch Write Port Number

DNS	Domain Name Service
Doom	Doom, the game
DPA	Microsoft's Distributed Password Authentication
DRP	DECnet Routing Protocol
EGP	Network Routing Information (Exterior Gateway Protocol)
EIGRP	Enhanced Interior Gateway Routing Protocol
FileMaker Pro	Database Application
Finger	Finger User Information Protocol
FIX	Financial Information eXchange
FNA	Fujitsu Network Architecture (a variant of SNA)
FNAonTCP-1	Transport Independent Convergence - FNA on TCP port 492
FNAonTCP-2	Transport Independent Convergence - FNA on TCP port 493
FTP	File Transfer Protocol service group classification—both FTP commands and data
FTP-Cmd	File Transfer Protocol command channel
FTP-Data	File Transfer Protocol data transfer channel
Gopher	Search application
GRE	General Routing Encapsulation
Groupwise	Novell Groupwise messaging system
Groupwise-MTA	Novell Groupwise Message Transfer Agent
Groupwise-POA	Novell Groupwise Post Office Agent
H.323	Internet telephony standard service group
H.323-GKD	H.323 Gatekeeper Discovery
H.323-H.245	H.323 call control
H.323-Q.931	H.323 call setup
H.323-RAS	H.323 Gatekeeper Control (Registration, Admission, and Status)
HTTP (Web)	Web traffic—Hypertext Transport Protocol
I-Phone	Phone service via the Internet
ICMP	Internet Control Message Protocol
Ident	Identification Protocol
IGMP	Internet Group Management Protocol
IMAP	Interactive Mail Access Protocol
IP	Internet Protocol (not auto-discovered)
IPSec	IP Security Encapsulation
IPSec-AH	IPSec Authentication Header
IPSec-ESP	IPSec Encapsulating Security Payload
IPv6	Internet Protocol version 6
IPX	Novell's networking protocol
IRC	Internet Relay Chat
IRC-194	IRC on port 194
IRC-6665	IRC on port 6665 (server to server)
IRC-6667	IRC on port 6667 (client to server)

ISAKMP	ISAKMP/IKE key exchange
Kali	Gaming Protocol
Kerberos	Network Authentication Service (ticket granting and checking)
L2TP	Level 2 Tunneling Protocol for VPN connections (UDP encapsulation)
LAT	DEC Printer Support (Local Area Transport)
LDAP	Lightweight Directory Access Protocol
Lockd	Lock Daemon
LotusNotes	Groupware for collaborative communication
Marimba	Marimba's Castanet push technology
Micom-VIP	Micom Voice over IP (V/IP)
MPEG-Audio	Moving Picture Experts Group - Audio Streams
MPEG-Video	Moving Picture Experts Group - Video Streams
MSSQ	Microsoft Message Queue Traffic
MSSQ-CQ	MSSQ Client Queue
MSSQ-IS	MSSQ Information Store
MSSQ-Ping	MSSQ Ping Mechanism
MSSQ-QMT	MSSQ Queue Manager Traffic
MSSQ-SQ	MSSQ Server Queue
MS-SQL	Service group for both Microsoft SQL Mon and Server traffic
MS-SQL-Mon	Microsoft SQL Monitor
MS-SQL-Server	Microsoft Structured Query Language (SQL) Server
NetBEUI	Service group for NetBEUI—Network protocol for PCs
NetBIOS-IP	NetBIOS over IP
NetBIOS-IP-NS	NetBIOS Name Service
NetBIOS-IP-SSN	NetBIOS Session Service
NFS	Network File System (both TCP and UDP)
NNTP (News)	Network News Transfer Protocol
NTP	Network Time Protocol
NW5-CMD	Netware 5 - Compatibility Mode Drivers service group
NW5-CMD-TCP	Netware 5 - Compatibility Mode Drivers over TCP
NW5-CMD-UDP	Netware 5 - Compatibility Mode Drivers over UDP
NW5-NCP	Netware 5 Core Protocol
OpenConnect-JCP	Browser-based access to host applications
Oracle	Database application
OracleClient	Oracle Java client (Webforms)
Oracle-netv1	Oracle SQL*Net v1
Oracle-netv2	Oracle SQL*Net v2
Oracle 8i	Oracle 8i by database name
OSI	OSI over TCP (RFC2126), e.g., Microsoft Exchange X.400
OSPF	Network Routing Information (Open Shortest-Path First)
pcAnywhere	Remote management collaboration tool

pcAnywhere-D	pcAnywhere data
pcAnywhere-OD	pcAnywhere data (old port)
pcAnywhere-OS	pcAnywhere status (old port)
pcAnywhere-S	pcAnywhere status
PIM	Protocol-Independent Multicast Routing Protocol
PointCast	Push technology application
POP3 (Mail)	Post office protocol for email POP3 (Mail)
PPTP	Point-to-Point Tunneling Protocol
Printer	UNIX line printer spooler (LPR)
Quake	Quake, the game
Quake-A	Quake 1
Quake-B	Quake 2
Quake-II-TCP	Quake over TCP
Quake-II-UDP	Quake over UDP
RADIUS	Service group for Remote Authentication Dial-in Service
RADIUS-Acct	RADIUS accounting service
RADIUS-Auth	RADIUS authentication service
RARP	Reverse Address Resolution Protocol
RC5DES	DES (data encryption standard) encryption-cracking application
RDP	Remote Desktop Protocol—Microsoft's Windows Terminal Server
RealAudio	Service group for RealAudio streaming audio/video application—both TCP and UDP
RealAudio-TCP	Streaming audio/video application TCP channel
RealAudio-UDP	Streaming audio/video application UDP channel
REXEC	UNIX remote execution protocol
RIP	Routing Information Protocol (UDP)
RTCP-B	Real-time control protocol (broadcast)
RTCP-I	Real-time control protocol (interactive)
RTP-B	Real-time protocol (broadcast)
RTP-I	Real-time protocol (interactive)
RTSP	Real-time Streaming Protocol
rwho	Reports current users for all hosts on the local network
SHOUTcast	Streaming audio
SLP	Service Location Protocol
SMS	Microsoft SMS (Systems Management Server) Help Desk
SMS-Auth	Microsoft SMS authentication
SMS-Chat	Microsoft SMS remote chat
SMS-File	Microsoft SMS file transfer
SMS-RC	Microsoft SMS remote control
SMTP (Mail)	Simple Mail Transfer Protocol
SNA	IBM's Systems Network Architecture protocol
SNMP	Service group for both Simple Network Management Protocol monitor and traps

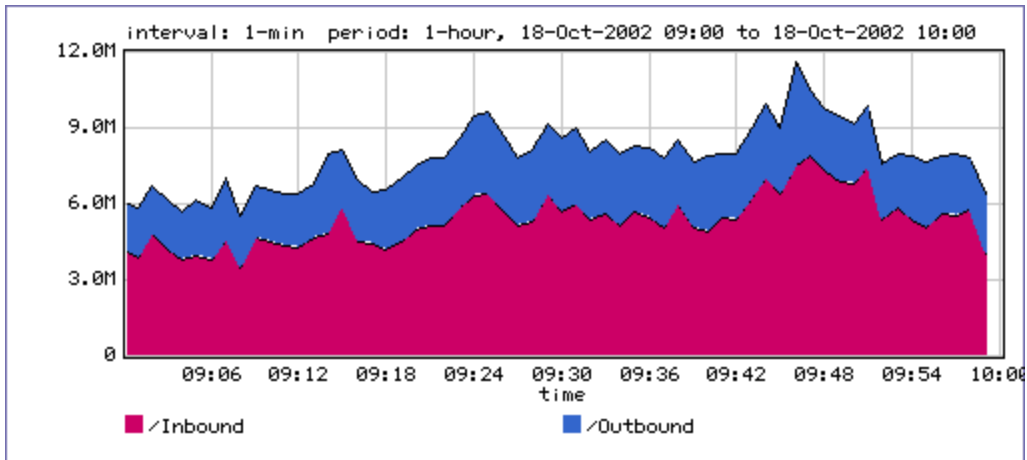
SNMP Mon	Simple Network Management Protocol monitor
SNMP Trap	Simple Network Management Protocol traps
SOCKS	SOCKS Proxy Protocol
Spanning Tree	IEEE802.1 Bridge Spanning Tree
SSH	Secure shell remote login protocol
SSL	Secure Sockets Layer protocol
ST2	Internet Stream Protocol, version 2
StreamWorks	StreamWorks Audio and Video
SunRPC	Sun's Remote Procedure Calls (UDP)
Syslog	UNIX System Logging
T.120	Collaboration application
TACACS	Login host protocol
TCP	Transmission Control Protocol—all Internet TCP traffic (not auto-discovered)
Telnet	Network terminal protocol
TFTP	Trivial File Transfer Protocol
Timbuktu	Timbuktu Pro service group; networked remote control application
Timbuktu-ctl	Timbuktu Control Channel
Timbuktu-hs	Timbuktu Handshaking
Timbuktu-obs	Timbuktu Observe Channel
Timbuktu-snd	Timbuktu Send Channel
Timbuktu-xch	Timbuktu Exchange Channel
TN3270	Telnet for IBM 3270 terminals and 3270 emulation
TN3287	IBM 3270 print traffic (TN3287 extensions)
TN5250	IBM 5250 terminal traffic over Telnet
TN5250p	IBM 5250 print traffic over Telnet
UDP	User Datagram Protocol—all Internet UDP traffic (not auto-discovered)
UUCP	Unix-to-Unix Copy Protocol
VDOPhone	Service group for Internet telephone application (not auto-discovered)
VDOPhone-a	Internet telephone application—TCP port 1 (not auto-discovered)
VDOPhone-b	Internet telephone application—TCP port 2 (not auto-discovered)
VDOPhone-UDP	VDOPhone real-time media (not auto-discovered)
Whois	Application that identifies the owner of a domain name
WindowsMedia	Microsoft Windows Media Player
WindowsMedia-T	Windows Media Streaming over TCP
WindowsMedia-U	Windows Media Streaming over UDP
WINS	Windows Internet Name Service
XWindows	X11 Windowing agent (UDP)
XWindows-DM	XWindows Display Manager (XDMCP)
XWindows-S	XWindows Server
YahooMsg	Yahoo! Messenger

Source: PacketShaper Reference Guide version 4.1

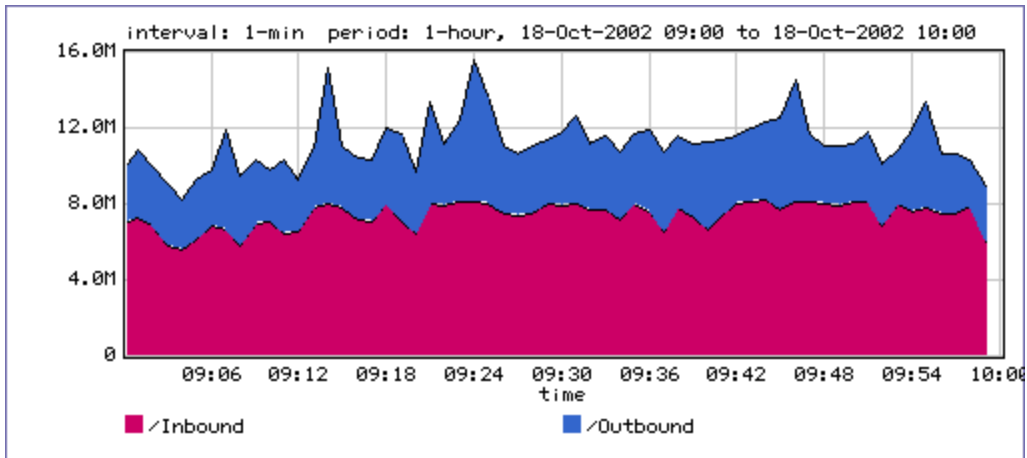
APPENDIX C

SAMPLE AVERAGE AND PEAK FLOW RATES

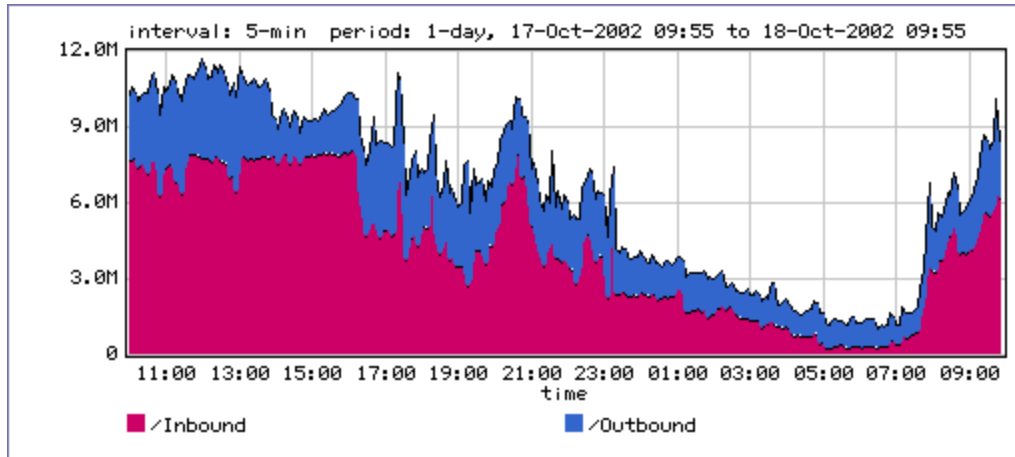
Hourly Average Rate (18 Oct 2002: 09:00 to 10:00)



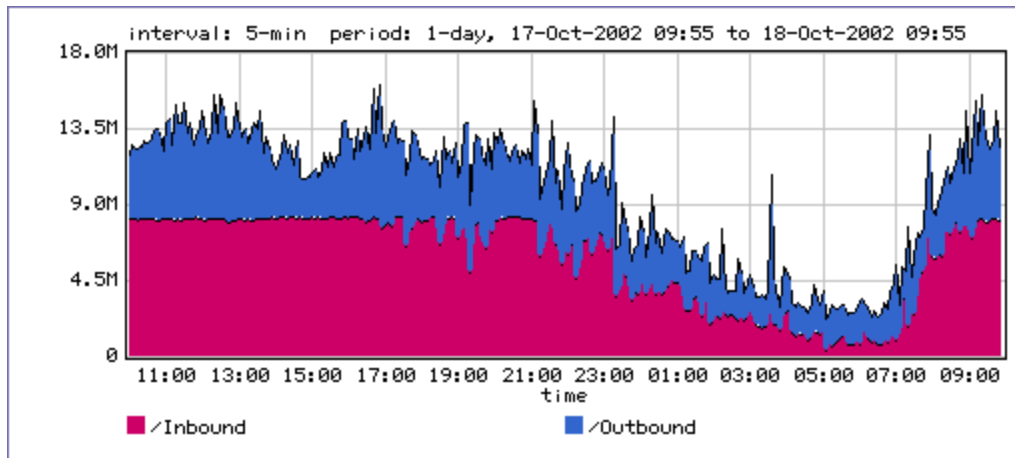
Hourly Peak Rate (18 Oct 2002: 09:00 to 10:00)



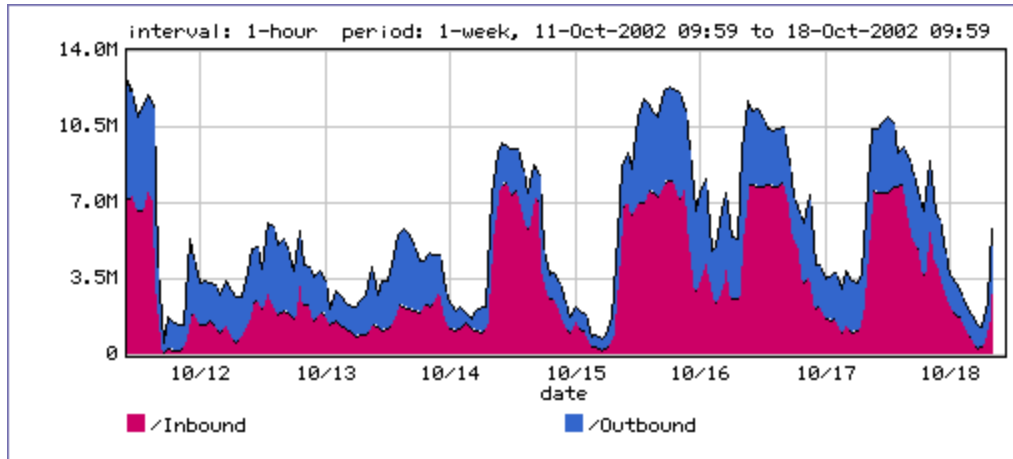
Daily Average Rate (17 Oct to 18 Oct 2002)



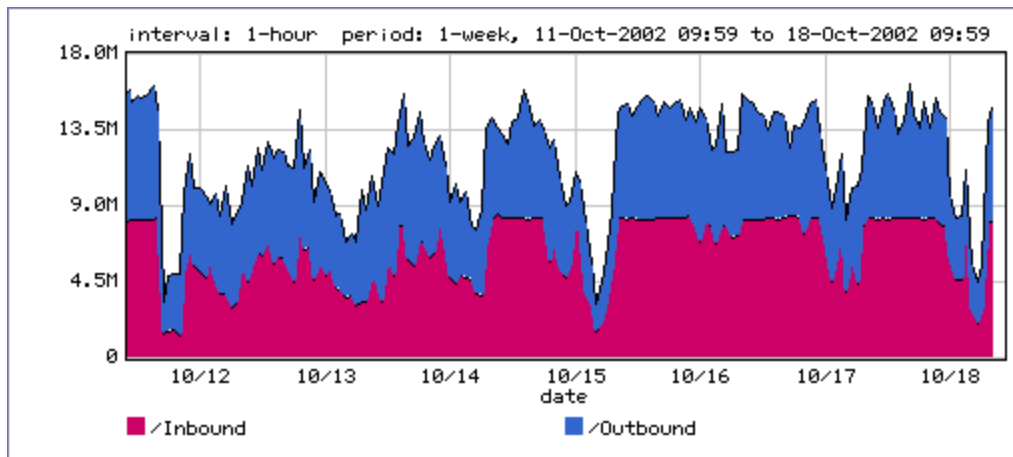
Daily Peak Rate (17 Oct to 18 Oct 2002)



Weekly Average Rate (11 Oct to 18 Oct 2002)



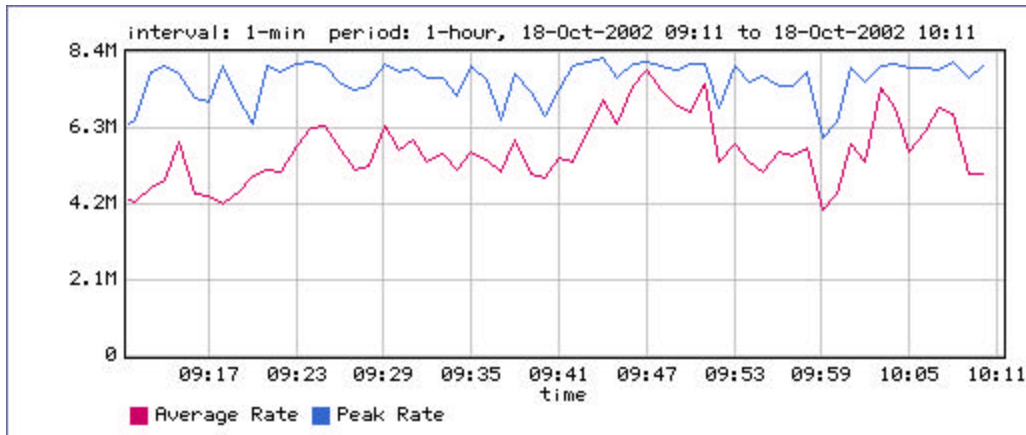
Weekly Peak Rate (11 Oct to 18 Oct 2002)



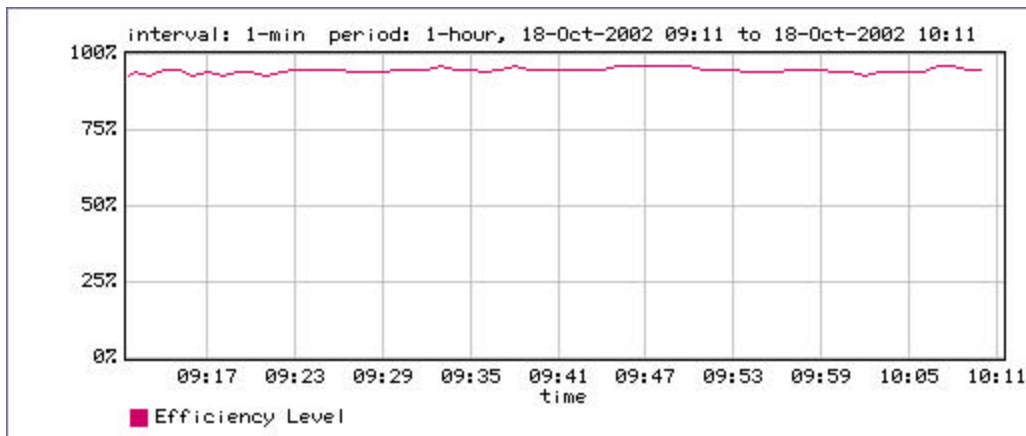
APPENDIX D

NETWORK PERFORMANCE SUMMARY

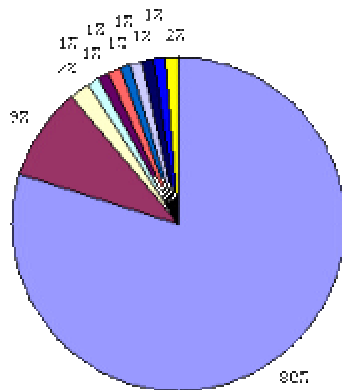
Hourly Inbound Utilization



Hourly Inbound Network Efficiency



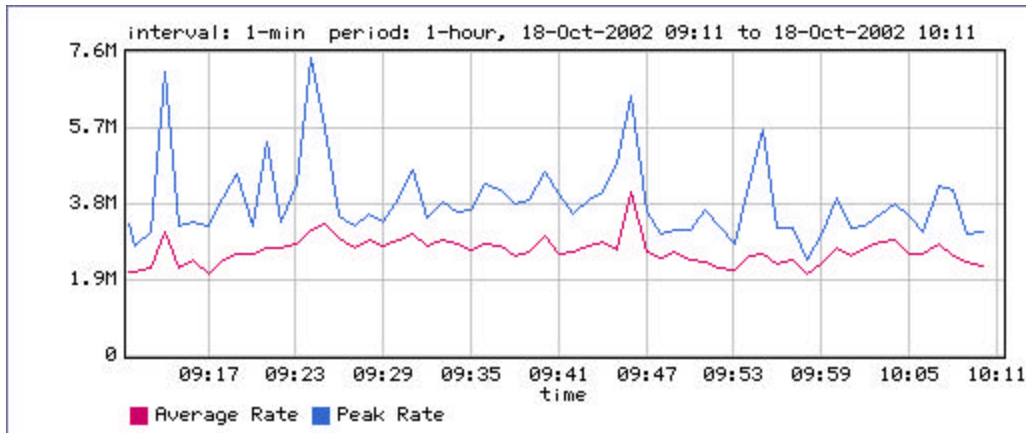
Hourly Inbound Top 10 Classes



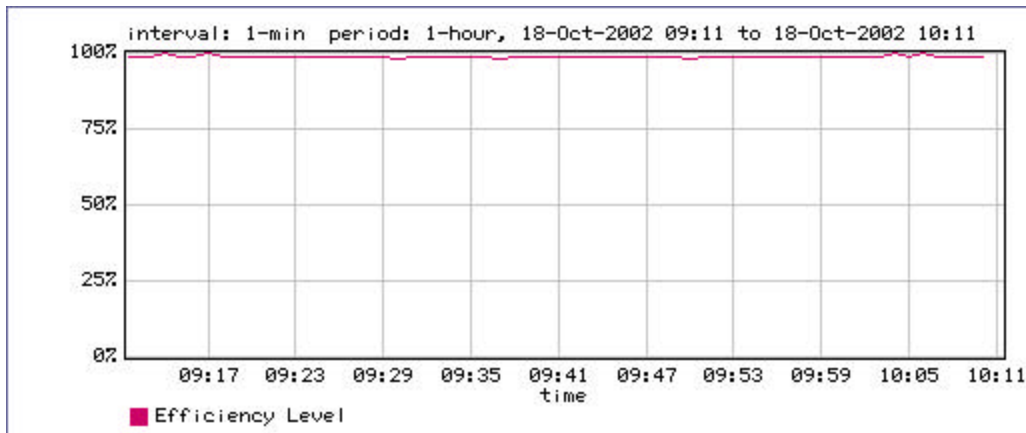
Class Name	Average Rate (bps)	(%)
1. /Inbound/HTTP	4.5M	80
2. /Inbound/AutoDiscovered/Default	490k	9
3. /Inbound/HiMedia	136k	2
4. /Inbound/QuickTime	74.5k	1
5. /Inbound/SSL	69.9k	1
6. /Inbound/Real	69.6k	1
7. /Inbound/SHTTP	68.7k	1
8. /Inbound/RTSP	57.9k	1
9. /Inbound/FTP	53.4k	1
10. /Inbound/Default	45.3k	1
All other classes	120k	2

period: 1-hour, 18-Oct-2002 09:11 to 18-Oct-2002 10:11

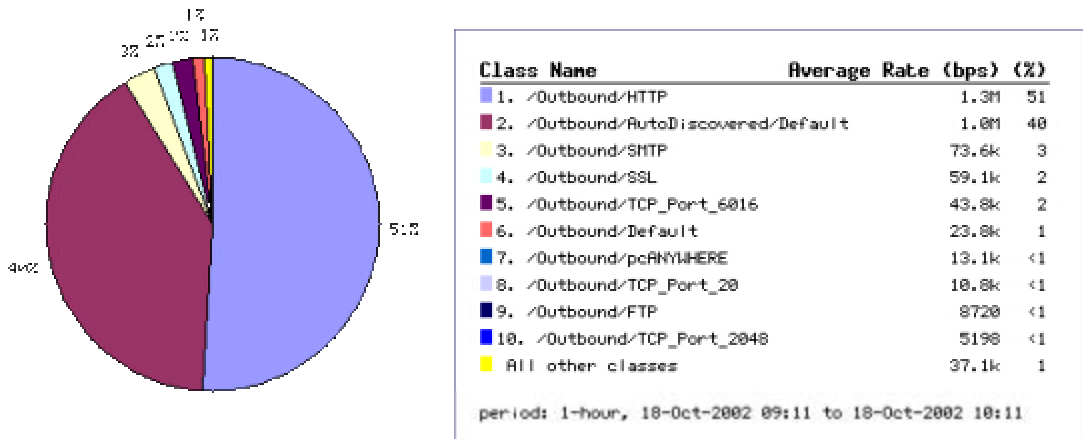
Hourly Outbound Utilization



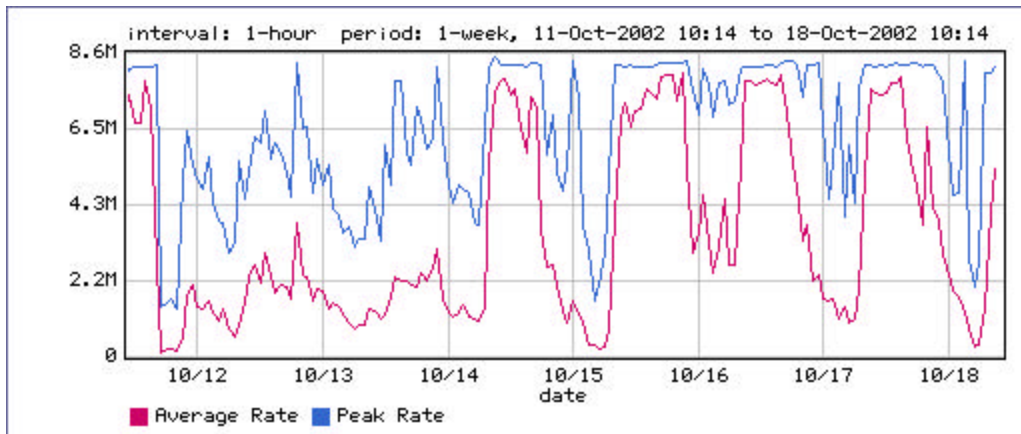
Hourly Outbound Network Efficiency



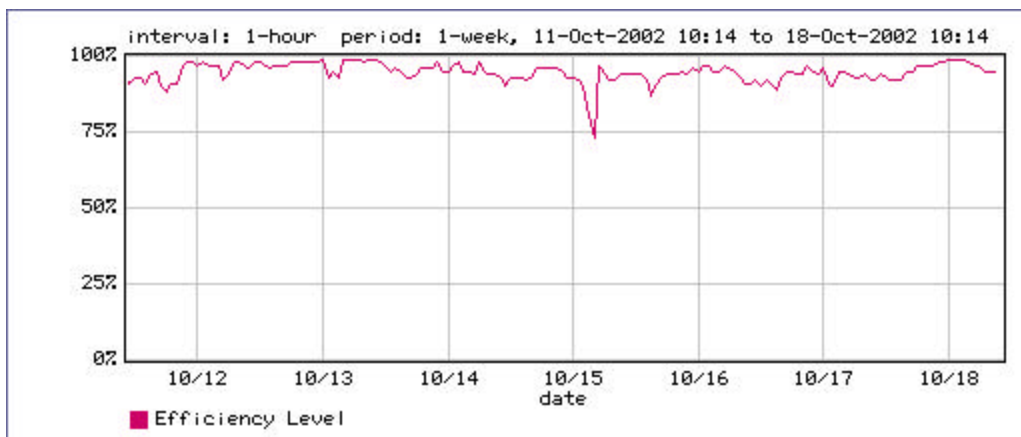
Hourly Outbound Top 10 Classes



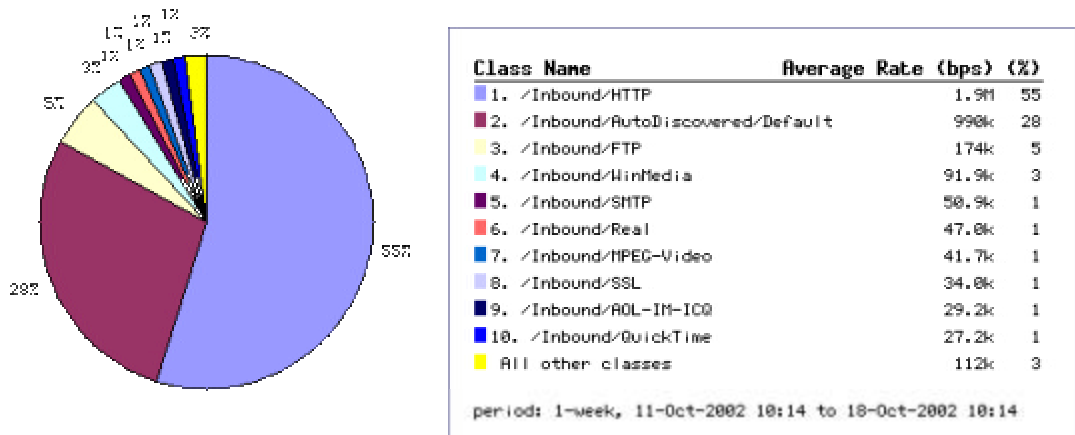
Weekly Inbound Utilization



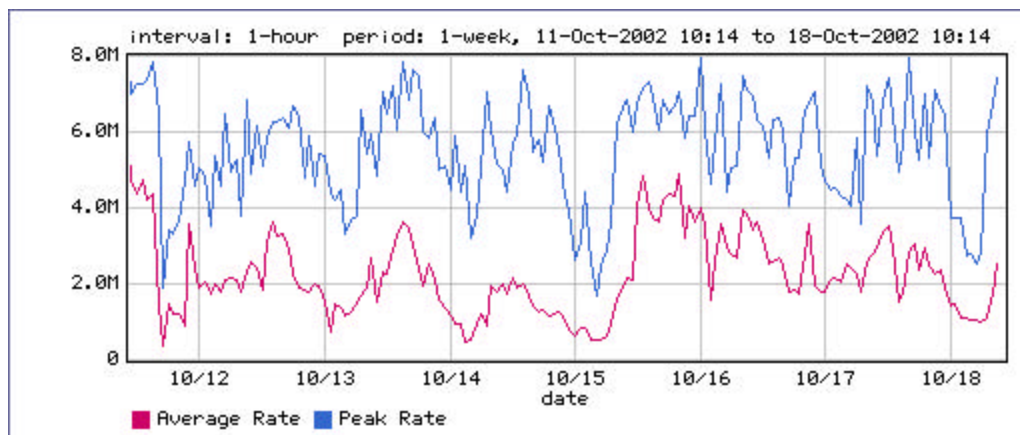
Weekly Inbound Network Efficiency



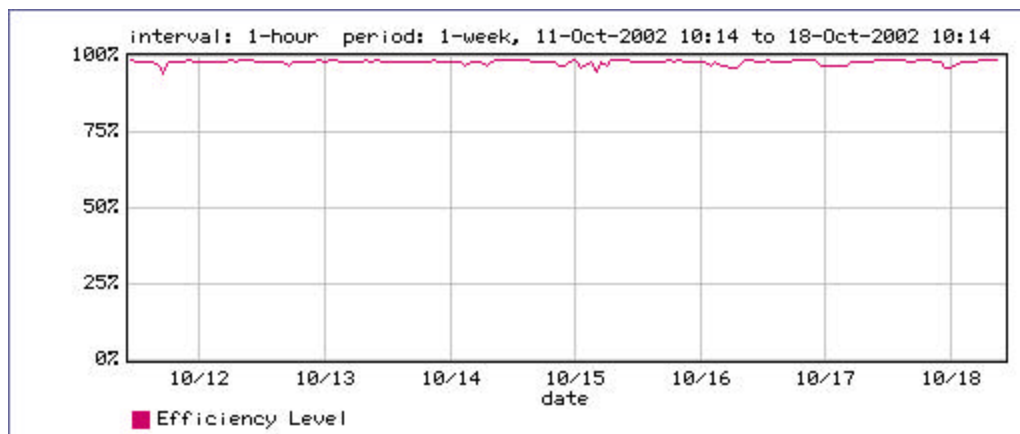
Weekly Inbound Top 10 Classes



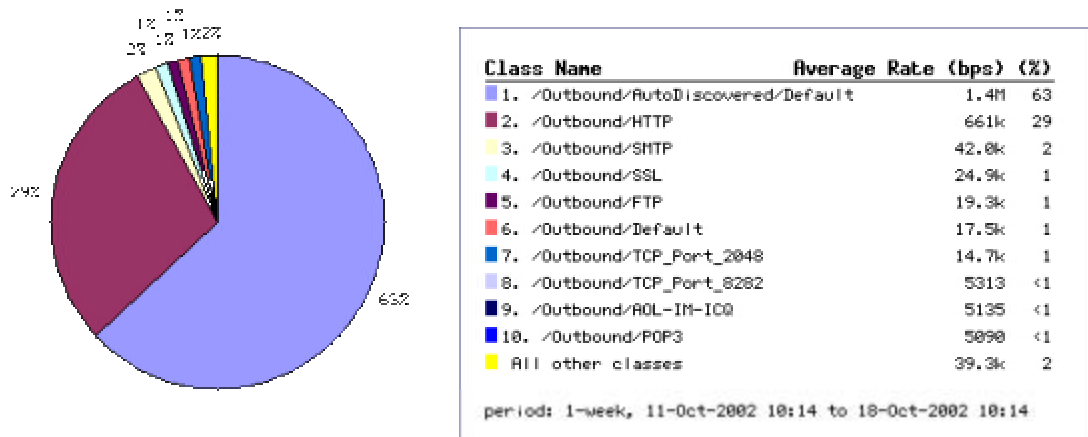
Weekly Outbound Utilization



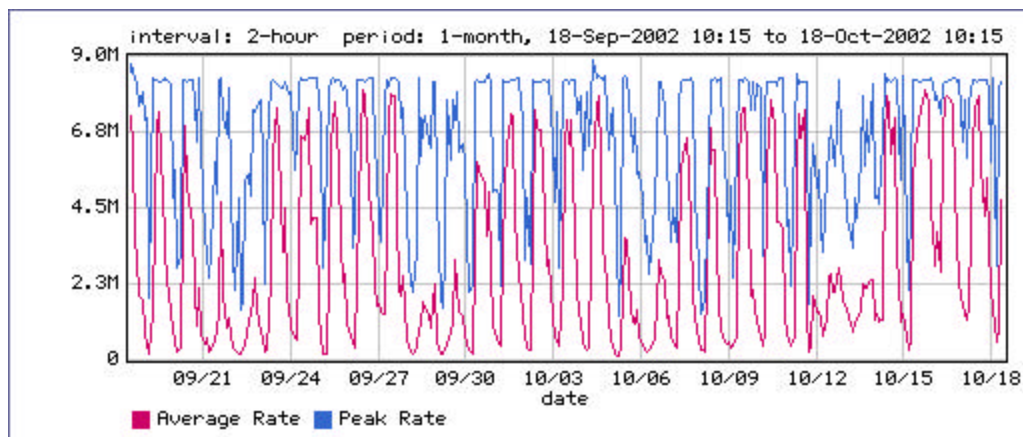
Weekly Outbound Network Efficiency



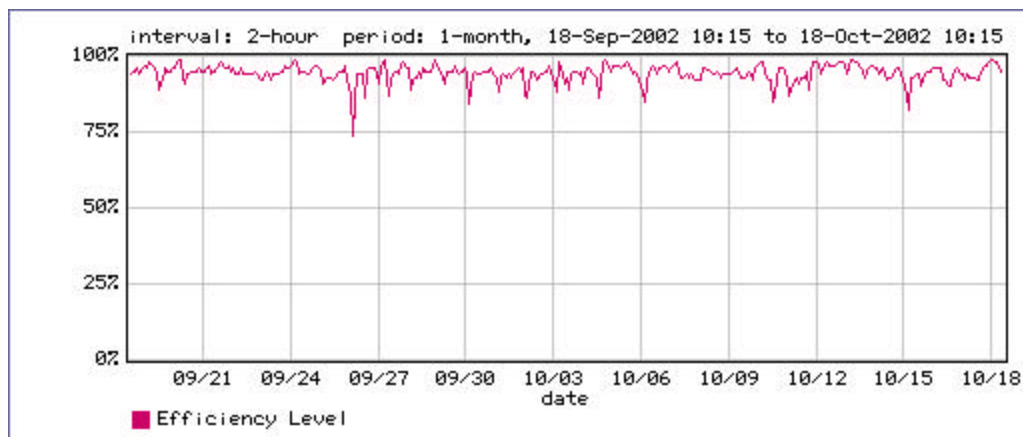
Weekly Outbound Top 10 Classes



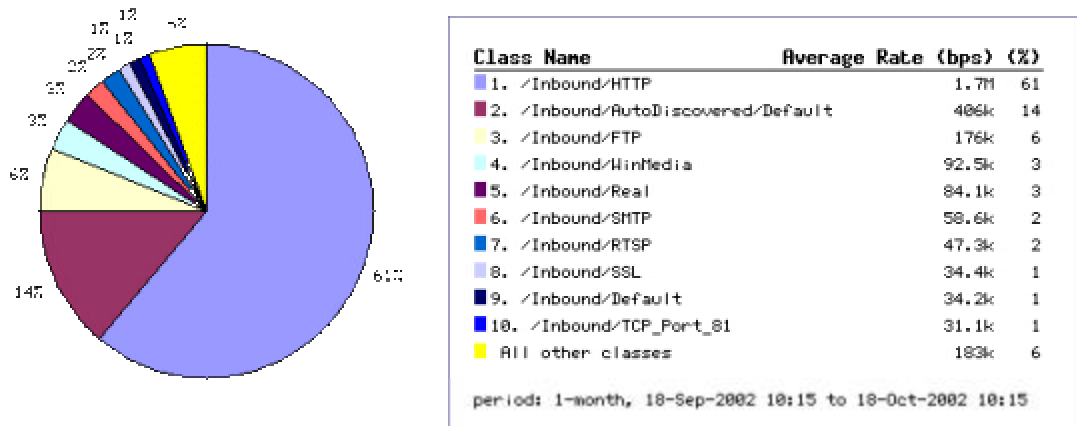
Monthly Inbound Utilization



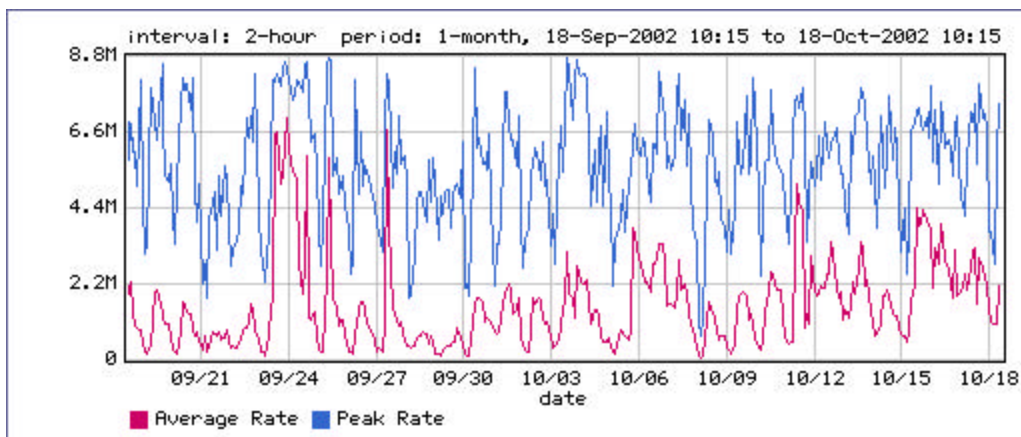
Monthly Inbound Network Efficiency



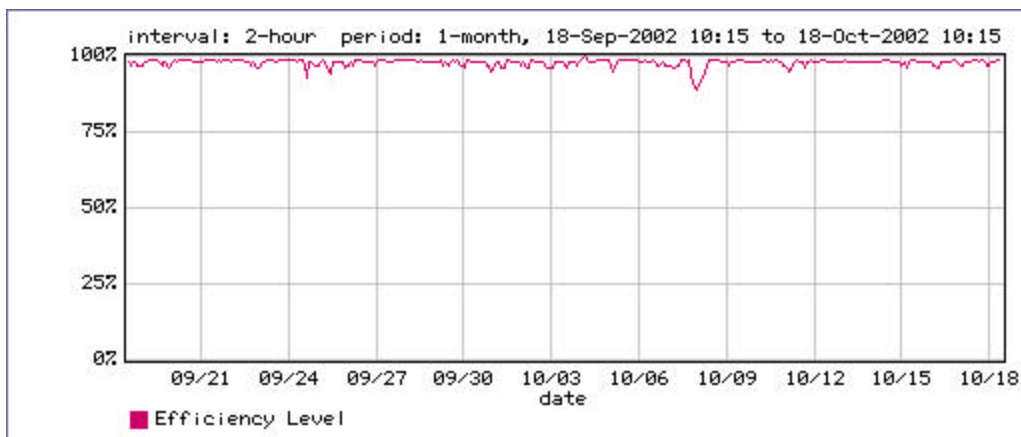
Monthly Inbound Top 10 Classes



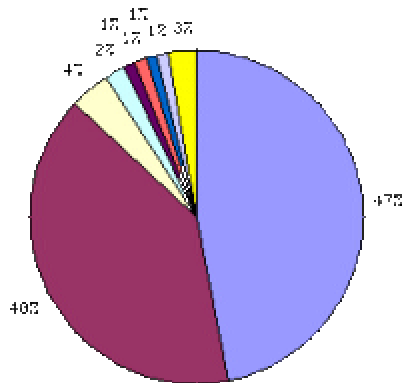
Monthly Outbound Utilization



Monthly Outbound Network Efficiency



Monthly Outbound Top 10 Classes



Class Name	Average Rate (bps)	(%)
1. /Outbound/AutoDiscovered/Default	734k	47
2. /Outbound/HTTP	631k	40
3. /Outbound/Default	59.4k	4
4. /Outbound/SNTP	33.7k	2
5. /Outbound/SSL	23.6k	1
6. /Outbound/ROL-IM-ICQ	15.7k	1
7. /Outbound/FTP	14.5k	1
8. /Outbound/TCP_Port_2048	10.6k	1
9. /Outbound/TCP_Port_8282	5927	<1
10. /Outbound/RDP	5328	<1
All other classes	55.5k	3

period: 1-month, 18-Sep-2002 10:16 to 18-Oct-2002 10:16

	Total Kbytes Received	Total Kbytes Sent
Hourly	2490145	1152978
Daily	44917399	21853781
Weekly	259153192	166159166
Monthly	905631746	497226101

APPENDIX E

ETSU NETWORK TRAFFIC TREE AS ON APRIL 01 2003

Class name	Type	Class	Policy	Cur	1 Min	Peak
		hits	hits	rate	avg	rate
/Inbound	+		n/a	5.3M	5.0M	8.9M
Localhost	PE	131	131	0	0	4
Discard	E		n/a	0	0	22
TCP_Port_4242	P	0	0	0	0	0
TCP_Port_6669	P	0	0	0	0	0
Audiogalaxy	P	0	0	0	0	0
CUSeeMe	P	0	0	0	0	0
Doom	P	0	0	0	0	0
eDonkey	P	0	0	0	0	0
Gnutella	P	26	26	0	0	0
Half-Life	P	0	0	0	0	0
IRC	P	0	0	0	0	0
KaZaA	P	2827	2827	0	0	0
Napster	P	4	4	0	0	0
Net2Phone	P	0	0	0	0	0
Quake	P	0	0	0	0	0
SHARESUDP	P	0	0	0	0	0
TFTP	P	0	0	0	0	0
Unreal	P	0	0	0	0	0
Webshots	P	3	3	0	0	10
YahooGames	P	6	6	0	0	0
Battle.net	P	0	0	0	0	0
Blubster	P	0	0	0	0	0
CU-DEV	P	0	0	0	0	0
Echo	P	0	0	0	0	0
Mythic	P	0	0	0	0	0
SonyOnline	P	0	0	0	0	0
High	E		n/a	8580	11.4k	123k
Kerberos	P	38	38	0	0	9906
LDAP	P	12	12	0	0	10
NTP	P	3468	3468	0	14	954
OracleEM	P	0	0	0	0	0
RADIUS	P	0	0	0	0	0
SNMP	P	5774	5774	0	577	10.9k
Day-Time	P	21	21	70	36	7921
DNS	P	158820	158820	8439	10.4k	122k
INFOC-RTMS	P	0	0	0	0	0
SLP	P	0	0	0	0	0

SMS	P	126	126	442	324	6427
TimeServer	P	36	36	2	2	397
WINS	P	0	0	0	0	0
Low	E	n/a	703k	714k	8.9M	
TCP_Port_10110	P	0	0	0	0	0
TCP_Port_10120	P	0	0	0	0	0
TCP_Port_1026	P	0	0	0	0	0
TCP_Port_2011	P	0	0	0	0	0
TCP_Port_5001	P	2	2	0	5	25.9k
TCP_Port_5010	P	0	0	0	0	0
TCP_Port_6009	P	42	42	493	272	3358
TCP_Port_6016	P	220	220	0	68	10.8k
TCP_Port_7226	P	0	0	0	0	0
TCP_Port_80	P	788	785	413	124	62.9k
TCP_Port_85	P	0	1	0	0	0
UDP_Port_6970	P	141	138	173k	197k	1.3M
AFS	P	0	0	0	0	0
BackWeb	P	4	4	0	0	21.5k
Chaincast	P	0	0	0	0	0
CIFS-TCP	P	0	0	0	0	0
CVSpserver	P	0	0	0	0	0
Finger	P	0	0	0	0	0
FTP	P	5312	5306	28.4k	18.3k	1.4M
Gopher	P	3	3	0	0	168
lockd	P	1	1	0	0	0
Marimba	P	2	2	18	18	18
MPEG-Audio	P	25	25	1	1	2.5M
MPEG-Video	P	161	161	2	2	2.9M
MSN-Messenger	P	191	190	149	67	52.7k
QuickTime	P	7	7	0	1805	1.0M
RC5DES	P	9	9	0	0	362
Real	P	257	257	80.5k	92.5k	2.2M
rsync	P	0	0	0	0	0
Shoutcast	P	0	0	0	0	0
SMTP	P	31994	31983	133k	112k	1.2M
WHOIS	P	1	1	0	0	3
WinMedia	P	294	292	258k	257k	1.3M
YahooMsg	P	30	30	64	76	4112
AOL-IM-ICQ	P	137	137	605	421	26.7k
ISAKMP	P	1	1	0	0	2776
Kontiki	P	16	16	0	0	292
Microsoft -ds	P	119	119	6	6	8.8M
NetBIOS-IP	P	131	131	0	0	568
NFS	P	0	0	0	0	0
RTSP	P	26	24	14	20	1230
Average	E	n/a	4.6M	4.4M	7.2M	

GRE	P	0	0	0	0	0
TCP_Port_8080	P	226	226	0	2	2851
ATSTCP	P	0	0	0	0	0
ccMail	P	0	0	0	0	0
Citrix		n/a		0	0	1
Default	P	1	1	0	0	1
CitrixIMA	P	0	0	0	0	0
DLS	P	0	0	0	0	0
FileMaker	P	1	1	0	5	3196
Groupwise	P	22	22	10.8k	5983	123k
HTTP	P	594372	593910	4.1M	4.2M	7.1M
Ident	P	7	7	0	0	213
IMAP	P	224	253	0	26	87.6k
LotusNotes	P	0	0	0	0	0
MATIP	P	0	0	0	0	0
MCK-Signaling	P	0	0	0	0	0
MSSQL	P	1	1	0	0	0
NNTP	P	0	0	0	0	0
NW5-CMD	P	1	1	0	0	0
Oracle	P	84	84	0	3	29.9k
Persona	P	0	0	0	0	0
POP3	P	5598	5588	1127	1067	279k
PPTP	P	1	1	0	0	0
RDP	P	1	1	0	0	0
RTCP-I	P	0	0	0	0	0
RTP-I	P	0	0	0	0	0
SMTBF	P	1	1	3	3	3
SSH	P	14	14	0	4609	215k
SSL	P	10176	10168	363k	248k	1.1M
T.120	P	0	0	0	0	0
Telnet	P	43	43	0	452	26.4k
Timbuktu	P	1	1	0	0	189
VNC	P	1	1	0	22	475k
WebEx	P	0	0	0	0	0
XWindows	P	0	0	0	0	0
DCOM	P	22	22	32	9	15.2k
H.323	P	1	1	0	4	2327
pcANYWHERE	P	972	972	0	8	31.6k
ICMP	#NAME?	15395	15395	284	171	17.2k
rsh		0	n/a	0	1	636
MeetingMaker		0	n/a	0	0	0
NW5-NCP		9	n/a	0	0	0
StreamWorks		1	n/a	0	0	0
DiscoveredPorts			n/a	156	142	41.9k
TCP_Port_12968		0	n/a	0	0	0
TCP_Port_13311		0	n/a	0	0	0

TCP_Port_17037		0	n/a	0	0	0
TCP_Port_17202		0	n/a	0	0	0
TCP_Port_17801		0	n/a	0	0	0
TCP_Port_18511		0	n/a	0	0	0
TCP_Port_19397		0	n/a	0	0	0
TCP_Port_2048		287	n/a	100	83	24.3k
TCP_Port_21153		0	n/a	0	0	0
TCP_Port_2443		107	n/a	0	22	13.5k
TCP_Port_26161		0	n/a	0	0	0
TCP_Port_27423		0	n/a	0	0	0
TCP_Port_32110		0	n/a	0	0	0
TCP_Port_34277		0	n/a	0	0	0
TCP_Port_36935		0	n/a	0	0	0
TCP_Port_3899		0	n/a	0	0	0
TCP_Port_4000		0	n/a	0	0	0
TCP_Port_40376		0	n/a	0	0	0
TCP_Port_41320		0	n/a	0	0	0
TCP_Port_4170		0	n/a	0	0	0
TCP_Port_42783		0	n/a	0	0	0
TCP_Port_444		0	n/a	0	0	0
TCP_Port_45581		0	n/a	0	0	0
TCP_Port_46481		0	n/a	0	0	0
TCP_Port_46801		0	n/a	0	0	0
TCP_Port_5100		4	n/a	0	0	26.2k
TCP_Port_5190		34	n/a	101	73	15.4k
TCP_Port_5730		0	n/a	0	0	0
TCP_Port_59560		0	n/a	0	0	0
TCP_Port_63005		0	n/a	0	0	0
TCP_Port_7152		0	n/a	0	0	0
TCP_Port_7755		0	n/a	0	0	0
TCP_Port_8016		0	n/a	0	0	0
TCP_Port_8200		0	n/a	36	27	90
TCP_Port_8282		2137	n/a	56	79	31.7k
TCP_Port_8384		0	n/a	0	0	0
TCP_Port_88		14	n/a	0	0	11.5k
TCP_Port_8999		10	n/a	0	0	20.4k
TCP_Port_9217		0	n/a	0	0	0
TCP_Port_9991		0	n/a	0	0	0
UDP_Port_8197		0	n/a	0	0	0
Default	P I	18817		21420	1974	1869286k
/Outbound	+		n/a	970k	1.1M	9.1M
Localhost	PE	131		131	0	03
SameSide	PE	0		0	0	0
Discard	E		n/a		0	0
TCP_Port_4242	P	0		0	0	0

TCP_Port_6669	P	0	0	0	0	0
Audiogalaxy	P	0	0	0	0	0
CUSEeMe	P	0	0	0	0	0
Doom	P	0	0	0	0	0
eDonkey	P	0	0	0	0	0
Gnutella	P	30	30	0	0	0
Half-Life	P	0	0	0	0	0
IRC	P	0	0	0	0	0
KaZaA	P	4151	4151	0	0	0
MATIP	P	0	0	0	0	0
Napster	P	4	4	0	0	0
Net2Phone	P	0	0	0	0	0
Quake	P	0	0	0	0	0
Unreal	P	0	0	0	0	0
Webshots	P	1	1	0	0	0
YahooGames	P	6	6	0	0	0
Battle.net	P	0	0	0	0	0
Blubster	P	0	0	0	0	0
Mythic	P	0	0	0	0	0
SonyOnline	P	0	0	0	0	0
Average	E	n/a	888k	956k	6.3M	
GRE	P	0	0	0	0	0
ATSTCP	P	0	0	0	0	0
ccMail	P	0	0	0	0	0
Citrix		n/a		0	0	0
Default	P	1	1	0	0	0
CitrixIMA	P	0	0	0	0	0
DLS	P	0	0	0	0	0
FileMaker	P	1	1	0	9	1222
Groupwise	P	22	22	1694	1463	34.2k
HTTP	P	567265	566811	800k	884k	5.9M
Ident	P	7	7	0	0	317
IMAP	P	224	248	0	82	99.0k
LotusNotes	P	0	0	0	0	0
MCK-Signaling	P	0	0	0	0	0
MSSQL	P	1	1	0	0	66
NW5-CMD	P	1	1	0	0	0
Oracle	P	84	84	0	1	42.7k
Persona	P	0	0	0	0	0
POP3	P	5597	5587	2905	3019	636k
PPTP	P	1	1	0	0	107
RDP	P	1	1	0	0	65
RTCP-I	P	0	0	0	0	0
RTP-I	P	0	0	0	0	0
SMTBF	P	1	1	0	0	0
SSH	P	14	14	0	520	51.6k

SSL	P	10174	10166	55.4k	75.9k	1.5M
T.120	P	0	0	0	0	0
Telnet	P	43	43	0	526	236k
Timbuktu	P	1	1	0	0	97
VNC	P	1	1	0	39	20.5k
WebEx	P	0	0	0	0	0
XWindows	P	0	0	0	0	0
DCOM	P	22	22	31	9	14.3k
H.323	P	897	897	55	38	362
pcANYWHERE	P	971	971	0	0	66.6k
ICMP	#NAME?	35648	35648	253	428	112k
High	E		n/a	6589	6624	844k
DHCP	P	0	0	0	0	0
Kerberos	P	38	38	0	0	13.2k
LDAP	P	12	12	0	0	6
NTP	P	8391	8391	0	42	1606
OracleEM	P	0	0	0	0	0
RADIUS	P	0	0	0	0	0
SNMP	P	12281	12281	334	511	24.4k
Syslog	P	0	0	0	0	0
Day-Time	P	21	21	69	12	4572
DNS	P	184894	184894	6219	4360	843k
INFOC-RTMS	P	0	0	0	0	0
RIP	P	1	1	0	0	0
SLP	P	0	0	0	0	0
SMS	P	126	126	460	124	219k
TimeServer	P	36	36	2	2	416
WINS	P	0	0	0	0	0
RSVP	P	0	0	0	0	0
Low	E		n/a	95.2k	80.7k	9.1M
TCP_Port_10110	P	0	0	0	0	0
TCP_Port_10130	P	0	0	0	0	0
TCP_Port_17811	P	0	0	0	0	0
TCP_Port_2011	P	0	0	0	0	0
TCP_Port_5001	P	2	2	0	9	622
TCP_Port_5010	P	0	0	0	0	0
TCP_Port_85	P	0	0	0	0	0
UDP_Port_1235	P	0	0	0	0	0
UDP_Port_192	P	355	355	76	149	152
UDP_Port_5190	P	0	0	0	0	0
UDP_Port_6970	P	141	138	4854	4531	17.0k
AFS	P	0	0	0	0	0
BackWeb	P	4	4	0	0	3115
Chaincast	P	0	0	0	0	0
CIFS-TCP	P	0	0	0	0	0
CVSpserver	P	0	0	0	0	0

FTP	P	5300	5294	53.1k	31.7k	500k
Gopher	P	3	3	0	0	175
lockd	P	1	1	0	0	0
Marimba	P	2	2	26	26	26
MPEG-Audio	P	25	25	1	1	589k
MPEG-Video	P	161	161	0	5	47.6k
MSN-Messenger	P	191	190	202	73	7858
NNTP	P	0	0	0	0	0
QuickTime	P	6	6	0	70	14.4k
RC5DES	P	9	9	0	0	379
Real	P	254	254	2691	4880	34.7k
rsync	P	0	0	0	0	0
Shoutcast	P	0	0	0	0	0
SMTP	P	31992	31981	29.3k	25.9k	6.0M
SSDP	P	0	0	0	0	0
TFTP	P	960	960	0	45	8161
WHOIS	P	1	1	0	0	1
WinMedia	P	301	299	9153	9672	26.3k
YahooMsg	P	30	30	196	115	1287
AOL-IM-ICQ	P	137	137	551	245	14.0k
Echo	P	0	0	0	0	0
ISAKMP	P	1	1	0	0	4882
Kontiki	P	16	16	0	0	283
Microsoft-ds	P	119	119	0	5	9.0M
NetBIOS-IP	P	1507	1505	244	305	5794
NFS	P	1	1	0	0	0
RTSP	P	195	193	28	41	4109
SunRPC	P	31	31	0	0	4
Protocol_41	P	1	1	0	0	1
Protocol_54	P	54	54	0	0	577
Finger		0	n/a	0	0	0
rsh		0	n/a	0	3	1367
SHARESUDP		0	n/a	0	0	0
CU-DEV		0	n/a	0	0	0
MeetingMaker		0	n/a	0	0	0
NW5-NCP		9	n/a	0	0	220
StreamWorks		2	n/a	0	0	0
IPSec		1	n/a	0	0	1801
Protocol_255		0	n/a	0	0	0
DiscoveredPorts		n/a		939	795	429k
TCP_Port_10230		1	n/a	0	916	52.6k
TCP_Port_15126		0	n/a	0	0	0
TCP_Port_15533		0	n/a	0	0	0
TCP_Port_18429		0	n/a	0	0	0
TCP_Port_19052		0	n/a	0	0	0
TCP_Port_20169		0	n/a	0	0	0

TCP_Port_2048		287	n/a	51	42	147k
TCP_Port_2443		107	n/a	0	20	19.4k
TCP_Port_26161		0	n/a	0	0	0
TCP_Port_29798		3	n/a	0	1	6461
TCP_Port_3000		4	n/a	1	1	104
TCP_Port_34277		0	n/a	0	0	0
TCP_Port_36931		0	n/a	0	0	0
TCP_Port_37258		0	n/a	0	0	0
TCP_Port_3899		0	n/a	3	3	3
TCP_Port_42783		0	n/a	0	0	0
TCP_Port_46481		0	n/a	0	0	0
TCP_Port_46781		0	n/a	0	0	0
TCP_Port_47067		0	n/a	0	0	0
TCP_Port_50000		0	n/a	0	0	0
TCP_Port_5100		4	n/a	0	0	1468
TCP_Port_5190		34	n/a	97	78	2950
TCP_Port_6009		42	n/a	437	177	1578
TCP_Port_60090		0	n/a	0	0	0
TCP_Port_60132		0	n/a	0	0	0
TCP_Port_6016		220	n/a	360	101	13.9k
TCP_Port_6099		0	n/a	0	0	0
TCP_Port_63006		0	n/a	0	0	0
TCP_Port_7755		0	n/a	0	0	0
TCP_Port_8200		0	n/a	74	57	184
TCP_Port_8282		2137	n/a	28	40	429k
TCP_Port_8999		10	n/a	0	0	22.8k
TCP_Port_9324		0	n/a	0	0	0
TCP_Port_9909		0	n/a	0	0	0
UDP_Port_10615		0	n/a	0	0	0
UDP_Port_1130	P	0		0	0	0
UDP_Port_1138	P	0		0	0	0
UDP_Port_1320		0	n/a	0	0	0
UDP_Port_1622		0	n/a	0	0	0
UDP_Port_1678		0	n/a	0	0	0
UDP_Port_1920		0	n/a	0	0	0
UDP_Port_2174		0	n/a	0	0	0
UDP_Port_2312		0	n/a	0	0	0
UDP_Port_25		0	n/a	0	0	0
UDP_Port_2760		0	n/a	0	0	0
UDP_Port_2767		0	n/a	0	0	0
UDP_Port_2785		0	n/a	0	0	0
UDP_Port_2944		0	n/a	0	0	0
UDP_Port_3049		0	n/a	0	0	0
UDP_Port_3059		0	n/a	0	0	0
UDP_Port_3283		0	n/a	0	0	0
UDP_Port_3403		0	n/a	0	0	0

UDP_Port_3568		0	n/a	0	0	0
UDP_Port_3594		0	n/a	0	0	0
UDP_Port_3918		0	n/a	0	0	0
UDP_Port_4015		0	n/a	0	0	0
UDP_Port_4473		0	n/a	0	0	0
UDP_Port_6971		0	n/a	0	0	0
UDP_Port_7001		19	n/a	0	1	479
UDP_Port_9		1	n/a	0	0	0
Default	P I	20286	23167	13.9k	6377	2.9M

APPENDIX F

PROTOCOLS AND SERVICES RECOGNIZED BY PACKETPUP

Category	Specific Instances
FTP	FTPActive FTPControl FTPPassive
HTTP	HTTP HTTP-ACTIVEX HTTP-CodeRed HTTP-CodeRedII HTTP-EXE HTTP-IDA HTTP-RAR HTTPS HTTP-Zip DoubleClick
A/V Stream	HTTP-AVI HTTP-Audio-MPEG HTTP-Video-MPEG HTTP-QuickTime HTTP-ShockWave-Flash RealMedia 1 and 2 ShoutCast WebRadio WindowsMedia
Peer-to-Peer	Aimster AudioGalaxyLogin AudioGalaxySearch AudioGalaxyDownloadReq AudioGalaxyXfer Blubster DirectConnect Gnutella Gnutella Web GnutellaXfer KaZaA KaZaAXfer Microsoft-DS NapsterXfer NetBIOX-SSN ScourExchange ScourExchangeXfer SongSpy

	SpinFrenzy
Remote Shell	Rlogin REXEC RSH SSH Telnet XWindows VNC
Other TCP	
Non-IP	
Other IP	Finger Gopher
Messaging	AIMLogin AIMXfer AIMMsg ICQLogin ICQMsg MSNMessengerLogin MSNMessengerXfer E-mail/Newsgroups IMAP IMAPS NNTP POP POP3S SMTP Hotmail YahooMail

APPENDIX G

SAMPLE DATA FROM PACKETPUP LOGFILE

Feb	21	9:26:58	packetpup[1188]	Parsed rule HTTP on line 66
Feb	21	9:26:58	packetpup[1188]	Finished reading rule file
Feb	21	9:26:58	packetpup[1188]	listening on network device \Device\NPF_{6F35C19A-8365-4E2F-8ACB-DFD263669DE4}
Feb	21	9:26:58	packetpup[1188]	Allocated 1048576 bytes for PCAP buffer.
Feb	21	9:26:58	packetpup[1188]	listening
Feb	21	9:26:58	packetpup[1188]	Packet matched rule SMTP: 206.46.170.98.34317 -> 151.141.8.105.25
Feb	21	9:26:58	packetpup[1188]	M SMTP,206.46.170.98,34317,151.141.8.105,25
Feb	21	9:26:59	packetpup[1188]	Packet matched rule POP3S: 151.141.12.204.2866 -> 151.141.8.105.995
Feb	21	9:26:59	packetpup[1188]	M POP3S,151.141.12.204,2866,151.141.8.105,995
Feb	21	9:26:59	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6042
Feb	21	9:26:59	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6042
Feb	21	9:26:59	packetpup[1188]	Packet matched rule HTTPS: 151.141.48.22.1569 -> 151.141.8.105.443
Feb	21	9:26:59	packetpup[1188]	M HTTPS,151.141.48.22,1569,151.141.8.105,443
Feb	21	9:26:59	packetpup[1188]	Packet matched rule SMTP: 64.124.168.46.8766 -> 151.141.8.105.25
Feb	21	9:26:59	packetpup[1188]	M SMTP,64.124.168.46,8766,151.141.8.105,25
Feb	21	9:26:59	packetpup[1188]	Email/News Connection 64.124.168.46.8766->151.141.8.105.25 terminated 5.08 Packets/sec
Feb	21	9:26:59	packetpup[1188]	F Email/News,64.124.168.46,8766,151.141.8.105,25,12,2,0.15,25.08
Feb	21	9:27:00	packetpup[1188]	Packet matched rule SMTP: 151.141.99.22.1464 -> 151.141.8.105.25
Feb	21	9:27:00	packetpup[1188]	M SMTP,151.141.99.22,1464,151.141.8.105,25
Feb	21	9:27:00	packetpup[1188]	151.141.47.44.1215 151.141.8.105.80 mail.etsu.edu /exchange
Feb	21	9:27:00	packetpup[1188]	Packet matched rule POP3S: 151.141.31.77.4205 -> 151.141.8.105.995
Feb	21	9:27:00	packetpup[1188]	M POP3S,151.141.31.77,4205,151.141.8.105,995
Feb	21	9:27:01	packetpup[1188]	Packet matched rule SMTP: 151.141.8.103.25 -> 151.141.8.104.6044
Feb	21	9:27:01	packetpup[1188]	M SMTP,151.141.8.103,25,151.141.8.104,6044
Feb	21	9:27:01	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6047
Feb	21	9:27:01	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6047
Feb	21	9:27:02	packetpup[1188]	Packet matched rule HTTPS: 151.141.47.44.1216 -> 151.141.8.105.443
Feb	21	9:27:02	packetpup[1188]	M HTTPS,151.141.47.44,1216,151.141.8.105,443
Feb	21	9:27:02	packetpup[1188]	Packet matched rule HTTPS: 206.105.206.72.4710 -> 151.141.8.105.443
Feb	21	9:27:02	packetpup[1188]	M HTTPS,206.105.206.72,4710,151.141.8.105,443
Feb	21	9:27:03	packetpup[1188]	Packet matched rule SMTP: 151.141.99.22.1465 -> 151.141.8.105.25
Feb	21	9:27:03	packetpup[1188]	M SMTP,151.141.99.22,1465,151.141.8.105,25
Feb	21	9:27:03	packetpup[1188]	Packet matched rule SMTP: 151.141.99.22.1466 -> 151.141.8.105.25
Feb	21	9:27:03	packetpup[1188]	M SMTP,151.141.99.22,1466,151.141.8.105,25
Feb	21	9:27:03	packetpup[1188]	Packet matched rule HTTPS: 151.141.76.248.1044 -> 151.141.8.105.443
Feb	21	9:27:03	packetpup[1188]	M HTTPS,151.141.76.248,1044,151.141.8.105,443
Feb	21	9:27:04	packetpup[1188]	Packet matched rule SMTP: 151.141.8.103.25 -> 151.141.8.104.6049
Feb	21	9:27:04	packetpup[1188]	M SMTP,151.141.8.103,25,151.141.8.104,6049
Feb	21	9:27:06	packetpup[1188]	Packet matched rule SMTP: 206.106.119.10.4648 -> 151.141.8.105.25
Feb	21	9:27:06	packetpup[1188]	M SMTP,206.106.119.10,4648,151.141.8.105,25
Feb	21	9:27:06	packetpup[1188]	Packet matched rule SMTP: 24.106.95.18.2175 -> 151.141.8.105.25

Feb	21	9:27:06	packetpup[1188]	M SMTP,24.106.95.18,2175,151.141.8.105,25
Feb	21	9:27:06	packetpup[1188]	Packet matched rule POP3S: 151.141.41.212,2644 -> 151.141.8.105.995
Feb	21	9:27:06	packetpup[1188]	M POP3S,151.141.41.212,2644,151.141.8.105,995
Feb	21	9:27:06	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6053
Feb	21	9:27:06	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6053
Feb	21	9:27:06	packetpup[1188]	Packet matched rule IMAPS: 160.36.210.70.1073 -> 151.141.8.105.993
Feb	21	9:27:06	packetpup[1188]	M IMAPS,160.36.210.70,1073,151.141.8.105,993
Feb	21	9:27:06	packetpup[1188]	Packet matched rule IMAP: 151.141.8.103.143 -> 151.141.8.104.5373
Feb	21	9:27:06	packetpup[1188]	M IMAP,151.141.8.103,143,151.141.8.104,5373
Feb	21	9:27:06	packetpup[1188]	Packet matched rule HTTPS: 151.141.46.106.1347 -> 151.141.8.105.443
Feb	21	9:27:06	packetpup[1188]	M HTTPS,151.141.46.106,1347,151.141.8.105,443
Feb	21	9:27:07	packetpup[1188]	Packet matched rule HTTPS: 151.141.59.188.1107 -> 151.141.8.105.443
Feb	21	9:27:07	packetpup[1188]	M HTTPS,151.141.59.188,1107,151.141.8.105,443
Feb	21	9:27:08	packetpup[1188]	Packet matched rule SMTP: 216.39.115.52.4430 -> 151.141.8.105.25
Feb	21	9:27:08	packetpup[1188]	M SMTP,216.39.115.52,4430,151.141.8.105,25
Feb	21	9:27:11	packetpup[1188]	Packet matched rule HTTPS: 151.141.47.44.1217 -> 151.141.8.105.443
Feb	21	9:27:11	packetpup[1188]	M HTTPS,151.141.47.44,1217,151.141.8.105,443
Feb	21	9:27:11	packetpup[1188]	Packet matched rule HTTPS: 68.54.96.36.2081 -> 151.141.8.105.443
Feb	21	9:27:11	packetpup[1188]	M HTTPS,68.54.96.36,2081,151.141.8.105,443
Feb	21	9:27:11	packetpup[1188]	Packet matched rule HTTPS: 151.141.47.44.1218 -> 151.141.8.105.443
Feb	21	9:27:11	packetpup[1188]	M HTTPS,151.141.47.44,1218,151.141.8.105,443
Feb	21	9:27:12	packetpup[1188]	151.141.62.167.2010 151.141.8.105.80 mail.etsu.edu /default.asp
Feb	21	9:27:12	packetpup[1188]	Packet matched rule HTTPS: 151.141.76.248.1045 -> 151.141.8.105.443
Feb	21	9:27:12	packetpup[1188]	M HTTPS,151.141.76.248,1045,151.141.8.105,443
Feb	21	9:27:12	packetpup[1188]	Packet matched rule POP3S: 151.141.55.192.3319 -> 151.141.8.105.995
Feb	21	9:27:12	packetpup[1188]	M POP3S,151.141.55.192,3319,151.141.8.105,995
Feb	21	9:27:13	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6054
Feb	21	9:27:13	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6054
Feb	21	9:27:13	packetpup[1188]	Packet matched rule IMAPS: 151.141.8.57.63778 -> 151.141.8.105.993
Feb	21	9:27:13	packetpup[1188]	M IMAPS,151.141.8.57,63778,151.141.8.105,993
Feb	21	9:27:13	packetpup[1188]	Packet matched rule SMTP: 151.141.8.57.63777 -> 151.141.8.105.25
Feb	21	9:27:13	packetpup[1188]	M SMTP,151.141.8.57,63777,151.141.8.105,25
Feb	21	9:27:13	packetpup[1188]	Packet matched rule HTTPS: 24.159.43.221.4966 -> 151.141.8.105.443
Feb	21	9:27:13	packetpup[1188]	M HTTPS,24.159.43.221,4966,151.141.8.105,443
Feb	21	9:27:14	packetpup[1188]	Packet matched rule HTTPS: 151.141.68.167.2315 -> 151.141.8.105.443
Feb	21	9:27:14	packetpup[1188]	M HTTPS,151.141.68.167,2315,151.141.8.105,443
Feb	21	9:27:14	packetpup[1188]	Packet matched rule HTTPS: 151.141.62.167.2011 -> 151.141.8.105.443
Feb	21	9:27:14	packetpup[1188]	M HTTPS,151.141.62.167,2011,151.141.8.105,443
Feb	21	9:27:16	packetpup[1188]	Packet matched rule POP3S: 151.141.59.124.1132 -> 151.141.8.105.995
Feb	21	9:27:16	packetpup[1188]	M POP3S,151.141.59.124,1132,151.141.8.105,995
Feb	21	9:27:16	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6055
Feb	21	9:27:16	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6055
Feb	21	9:27:16	packetpup[1188]	Packet matched rule POP3S: 63.162.206.106.2865 -> 151.141.8.105.995
Feb	21	9:27:16	packetpup[1188]	M POP3S,63.162.206.106,2865,151.141.8.105,995
Feb	21	9:27:17	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6056
Feb	21	9:27:17	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6056

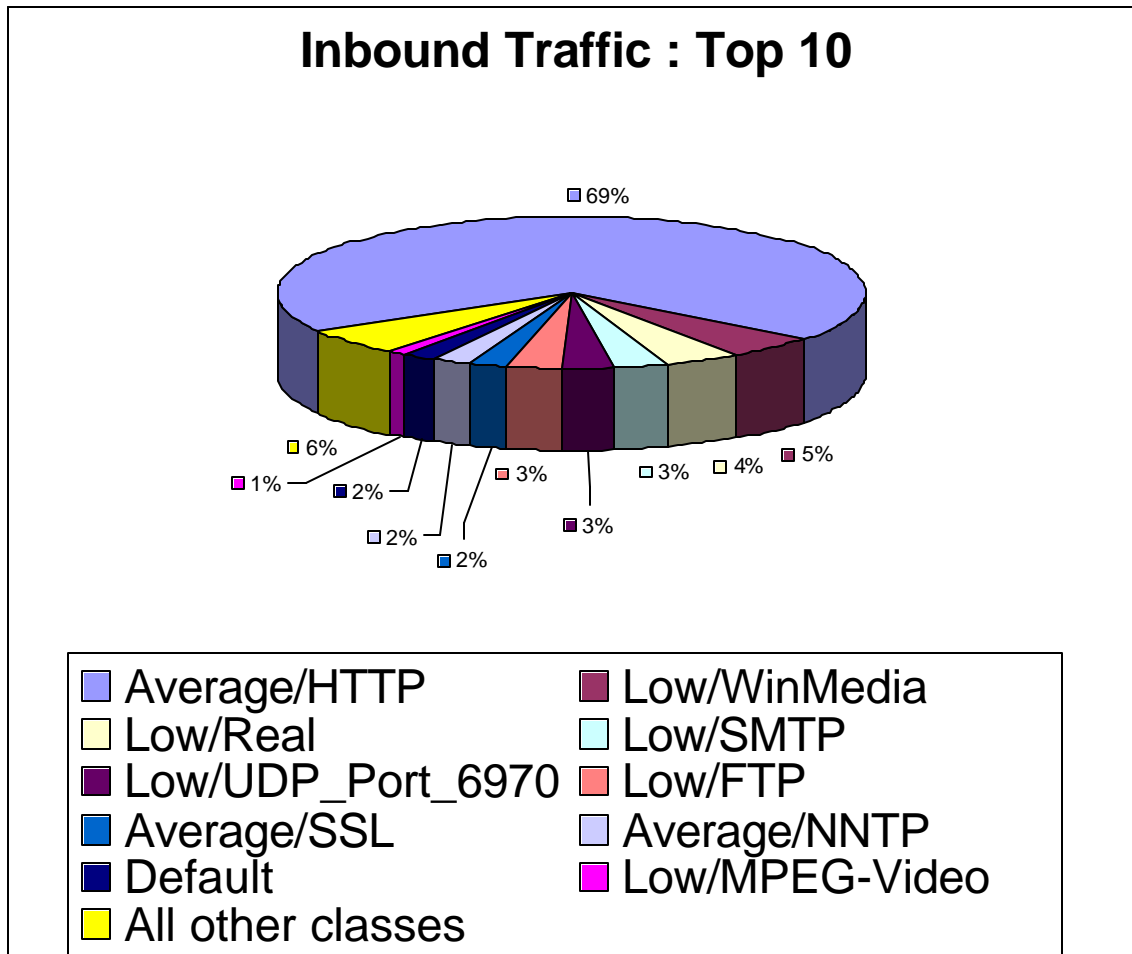
Feb	21	9:27:17	packetpup[1188]	Packet matched rule HTTPS: 151.141.47.44.1219 -> 151.141.8.105.443
Feb	21	9:27:17	packetpup[1188]	M HTTPS,151.141.47.44,1219,151.141.8.105,443
Feb	21	9:27:17	packetpup[1188]	Packet matched rule HTTPS: 151.141.46.106.1346 -> 151.141.8.105.443
Feb	21	9:27:17	packetpup[1188]	M HTTPS,151.141.46.106,1346,151.141.8.105,443
Feb	21	9:27:18	packetpup[1188]	WWW Connection 151.141.68.167.2315->151.141.8.105.443 terminated.63 Packets/sec
Feb	21	9:27:18	packetpup[1188]	F WWW,151.141.68.167,2315,151.141.8.105,443,1138,7,0.26,1.63
Feb	21	9:27:18	packetpup[1188]	Packet matched rule SMTP: 151.141.99.22.1469 -> 151.141.8.105.25
Feb	21	9:27:18	packetpup[1188]	M SMTP,151.141.99.22,1469,151.141.8.105,25
Feb	21	9:27:19	packetpup[1188]	Packet matched rule HTTPS: 151.141.68.167.2316 -> 151.141.8.105.443
Feb	21	9:27:19	packetpup[1188]	M HTTPS,151.141.68.167,2316,151.141.8.105,443
Feb	21	9:27:19	packetpup[1188]	Packet matched rule SMTP: 151.141.8.103.25 -> 151.141.8.104.6058
Feb	21	9:27:19	packetpup[1188]	M SMTP,151.141.8.103,25,151.141.8.104,6058
Feb	21	9:27:20	packetpup[1188]	Packet matched rule NNTP: 151.141.8.57.63796 -> 151.141.8.105.119
Feb	21	9:27:20	packetpup[1188]	M NNTP,151.141.8.57,63796,151.141.8.105,119
Feb	21	9:27:20	packetpup[1188]	Packet matched rule SMTP: 216.33.97.76.39843 -> 151.141.8.105.25
Feb	21	9:27:20	packetpup[1188]	M SMTP,216.33.97.76,39843,151.141.8.105,25
Feb	21	9:27:21	packetpup[1188]	Packet matched rule SMTP: 151.141.99.22.1470 -> 151.141.8.105.25
Feb	21	9:27:21	packetpup[1188]	M SMTP,151.141.99.22,1470,151.141.8.105,25
Feb	21	9:27:21	packetpup[1188]	Packet matched rule HTTPS: 65.162.104.178.1410 -> 151.141.8.105.443
Feb	21	9:27:21	packetpup[1188]	M HTTPS,65.162.104.178,1410,151.141.8.105,443
Feb	21	9:27:22	packetpup[1188]	Packet matched rule SMTP: 151.141.8.106.25 -> 151.141.8.104.6062
Feb	21	9:27:22	packetpup[1188]	M SMTP,151.141.8.106,25,151.141.8.104,6062
Feb	21	9:27:22	packetpup[1188]	Packet matched rule HTTPS: 151.141.62.167.2012 -> 151.141.8.105.443
Feb	21	9:27:22	packetpup[1188]	M HTTPS,151.141.62.167,2012,151.141.8.105,443
Feb	21	9:27:23	packetpup[1188]	Packet matched rule HTTPS: 151.141.62.167.2013 -> 151.141.8.105.443
Feb	21	9:27:23	packetpup[1188]	M HTTPS,151.141.62.167,2013,151.141.8.105,443
Feb	21	9:27:23	packetpup[1188]	WWW Connection 151.141.76.248.1044->151.141.8.105.443 terminated 30 Packets/sec
Feb	21	9:27:23	packetpup[1188]	F WWW,151.141.76.248,1044,151.141.8.105,443,714,6,0.04,0.30
Feb	21	9:27:23	packetpup[1188]	WWW Connection 151.141.76.248.1045->151.141.8.105.443 terminated.69 Packets/sec
Feb	21	9:27:23	packetpup[1188]	F WWW,151.141.76.248,1045,151.141.8.105,443,1113,8,0.09,0.69
Feb	21	9:27:23	packetpup[1188]	Packet matched rule POP3S: 24.158.97.34.21850 -> 151.141.8.105.995
Feb	21	9:27:23	packetpup[1188]	M POP3S,24.158.97.34,21850,151.141.8.105,995
Feb	21	9:27:24	packetpup[1188]	Packet matched rule POP: 151.141.8.103.110 -> 151.141.8.104.6065
Feb	21	9:27:24	packetpup[1188]	M POP,151.141.8.103,110,151.141.8.104,6065
Feb	21	9:27:24	packetpup[1188]	Packet matched rule HTTPS: 151.141.85.199.1171 -> 151.141.8.105.443
Feb	21	9:27:24	packetpup[1188]	M HTTPS,151.141.85.199,1171,151.141.8.105,443
Feb	21	9:27:24	packetpup[1188]	Packet matched rule HTTPS: 24.158.137.162.1378 -> 151.141.8.105.443
Feb	21	9:27:24	packetpup[1188]	M HTTPS,24.158.137.162,1378,151.141.8.105,443
Feb	21	9:27:24	packetpup[1188]	Packet matched rule SMTP: 151.141.8.106.25 -> 151.141.8.104.6067
Feb	21	9:27:24	packetpup[1188]	M SMTP,151.141.8.106,25,151.141.8.104,6067
Feb	21	9:27:25	packetpup[1188]	WWW Connection 24.158.137.162.1378->151.141.8.105.443 terminated 66 Packets/sec
Feb	21	9:27:25	packetpup[1188]	F WWW,24.158.137.162,1378,151.141.8.105,443,741,5,0.82,5.66
Feb	21	9:27:25	packetpup[1188]	Packet matched rule HTTPS: 24.158.137.162.1380 -> 151.141.8.105.443
Feb	21	9:27:25	packetpup[1188]	M HTTPS,24.158.137.162,1380,151.141.8.105,443
Feb	21	9:27:27	packetpup[1188]	WWW Connection 151.141.62.167.2012->151.141.8.105.443 terminated 5.80 Packets/sec
Feb	21	9:27:27	packetpup[1188]	F WWW,151.141.62.167,2012,151.141.8.105,443,2350,27,0.49,5.80

APPENDIX H

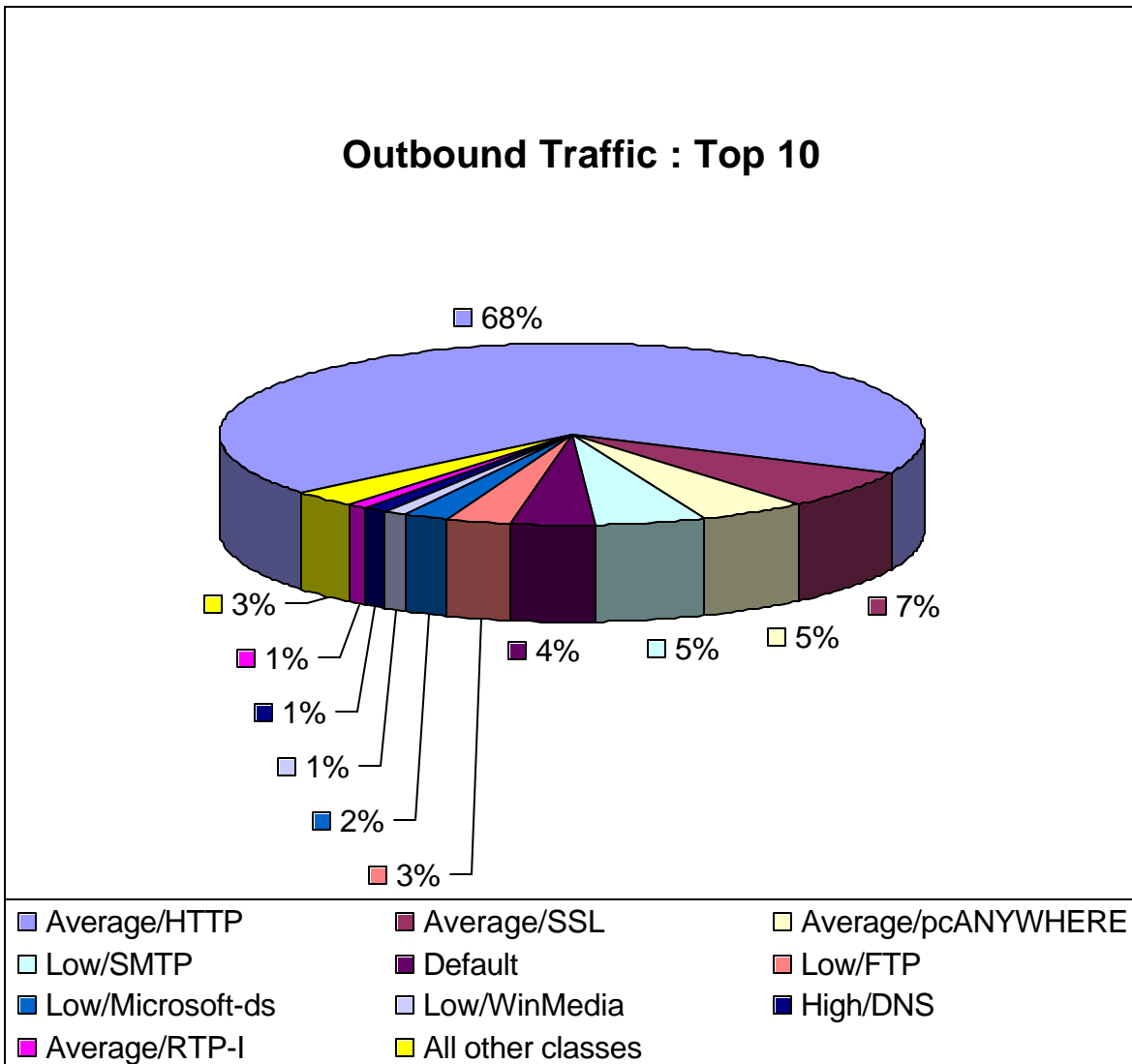
TOP TEN CLASS MEMBERS

Pie charts indicating the top 10 members of the different subclasses of the traffic tree categorized on the basis of the average flow rate are shown below.

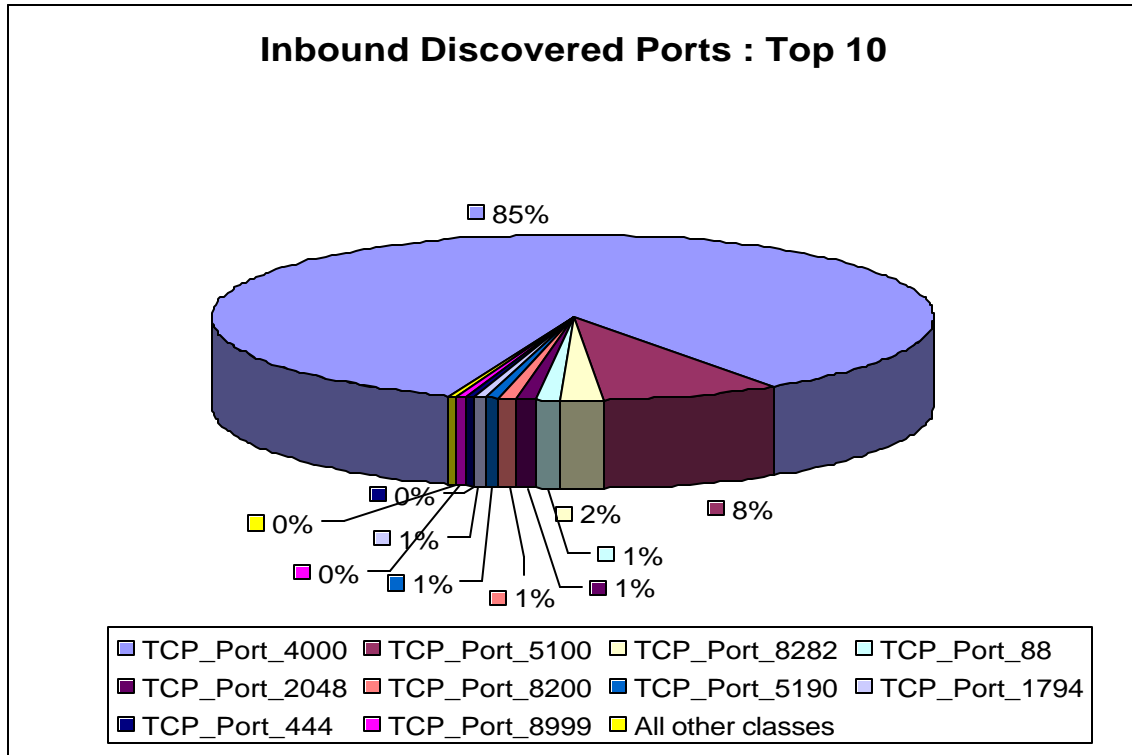
Inbound Traffic



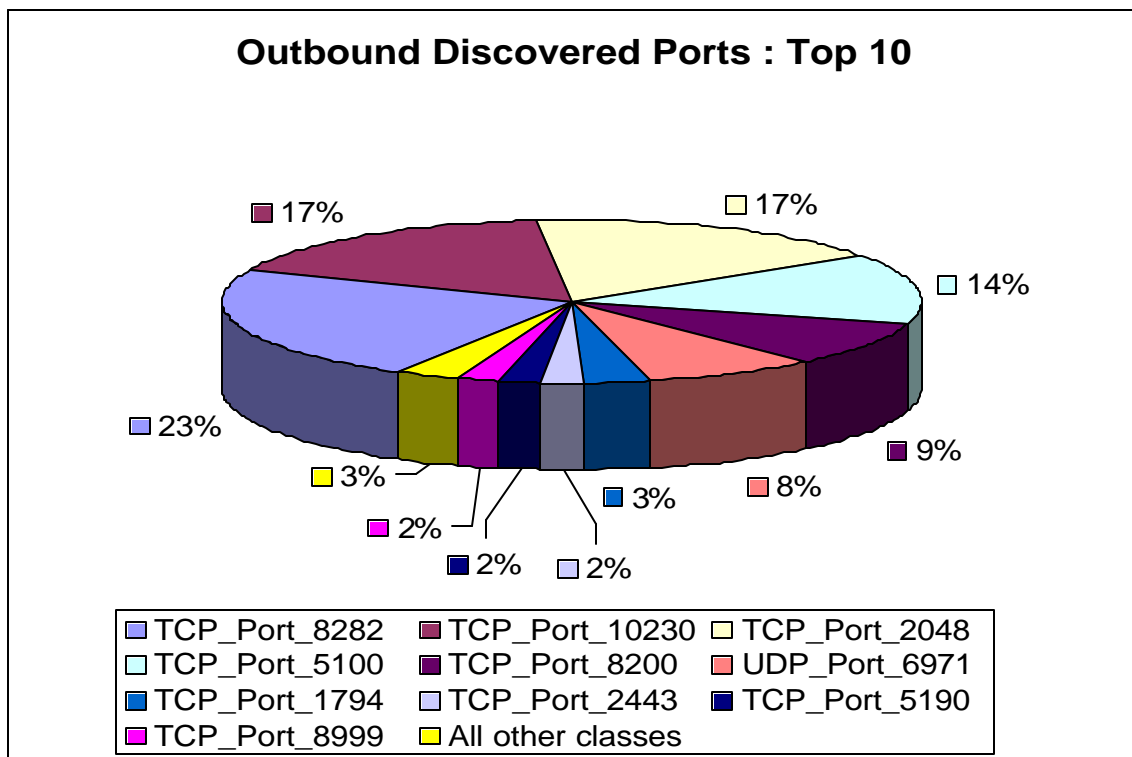
Outbound Traffic



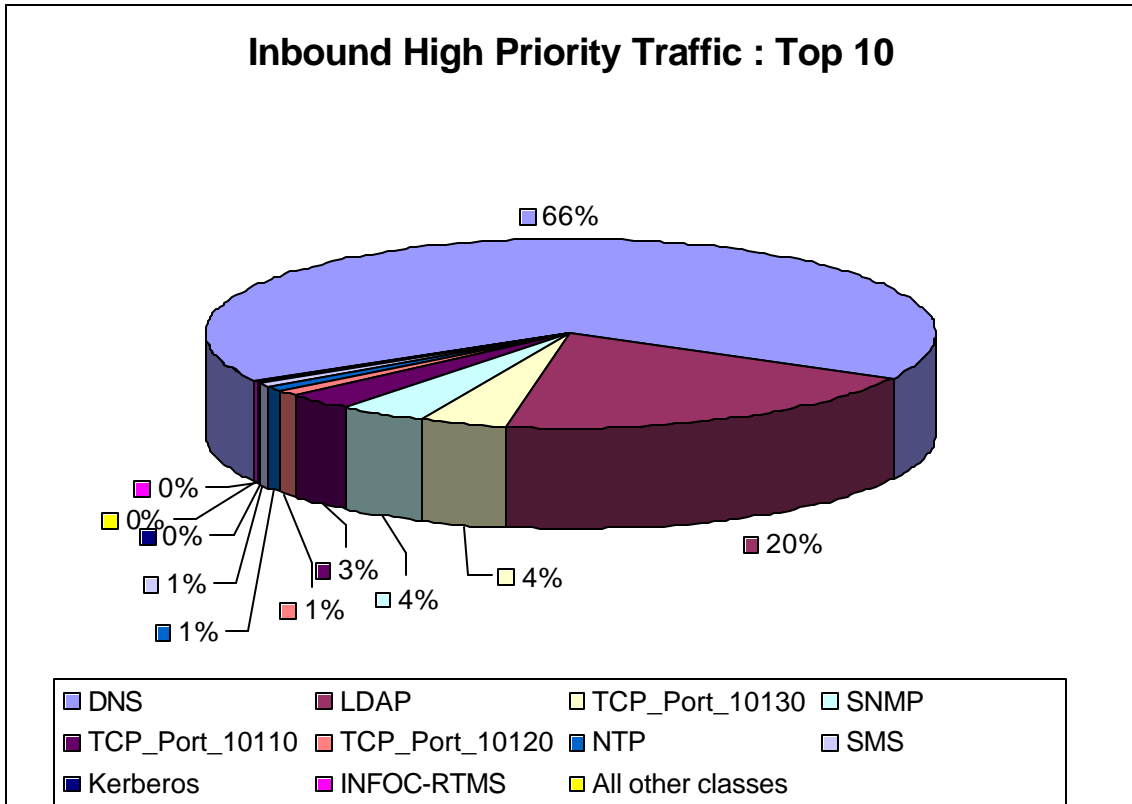
Inbound/Discovered Ports



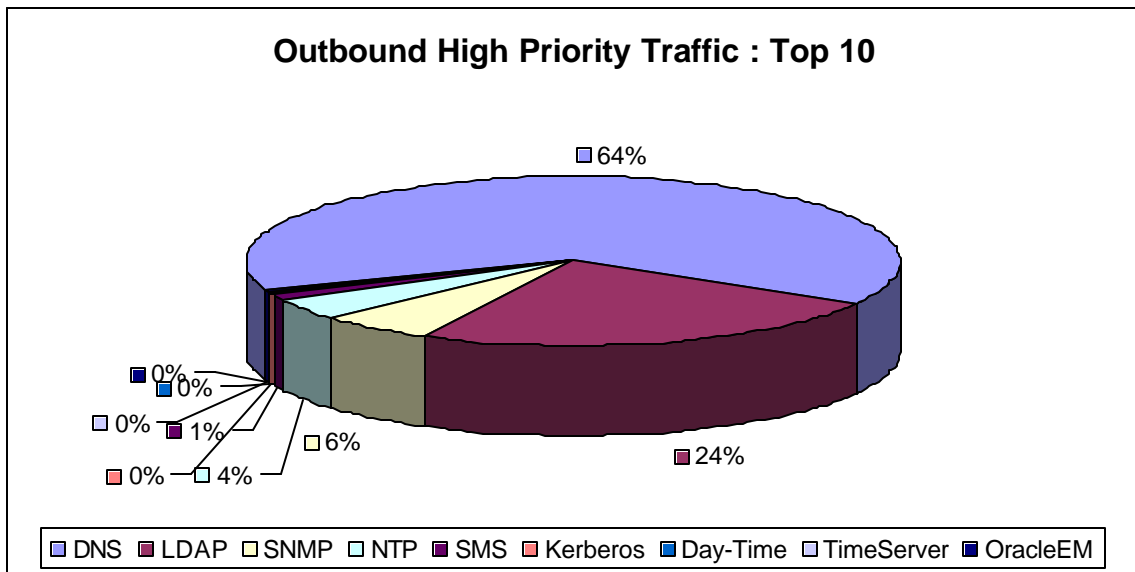
Outbound/Discovered Ports



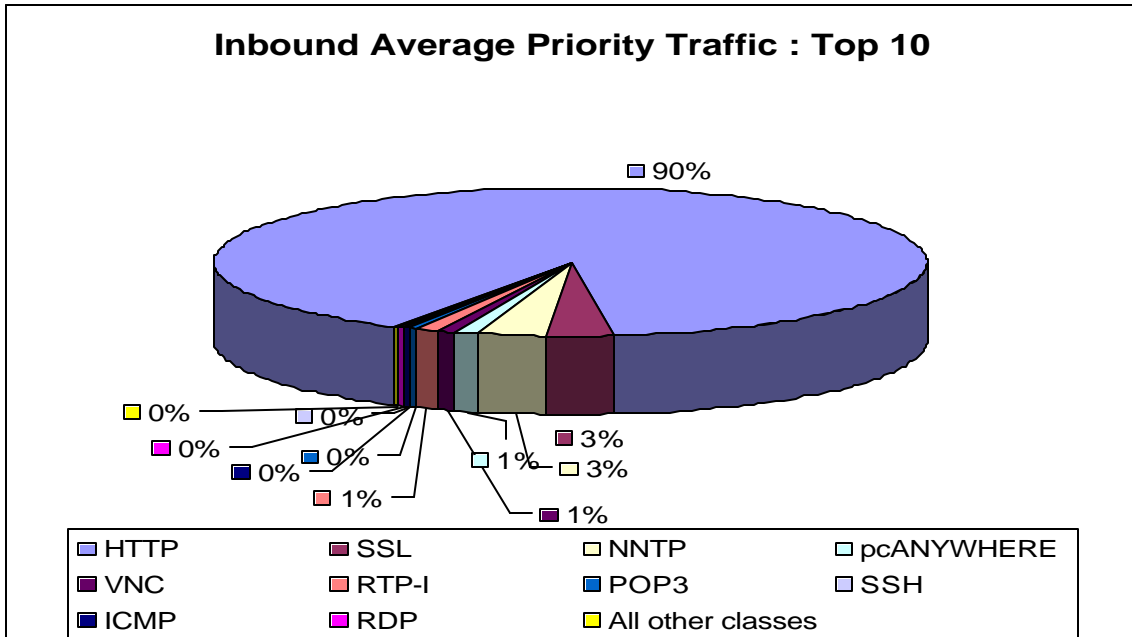
Inbound / High



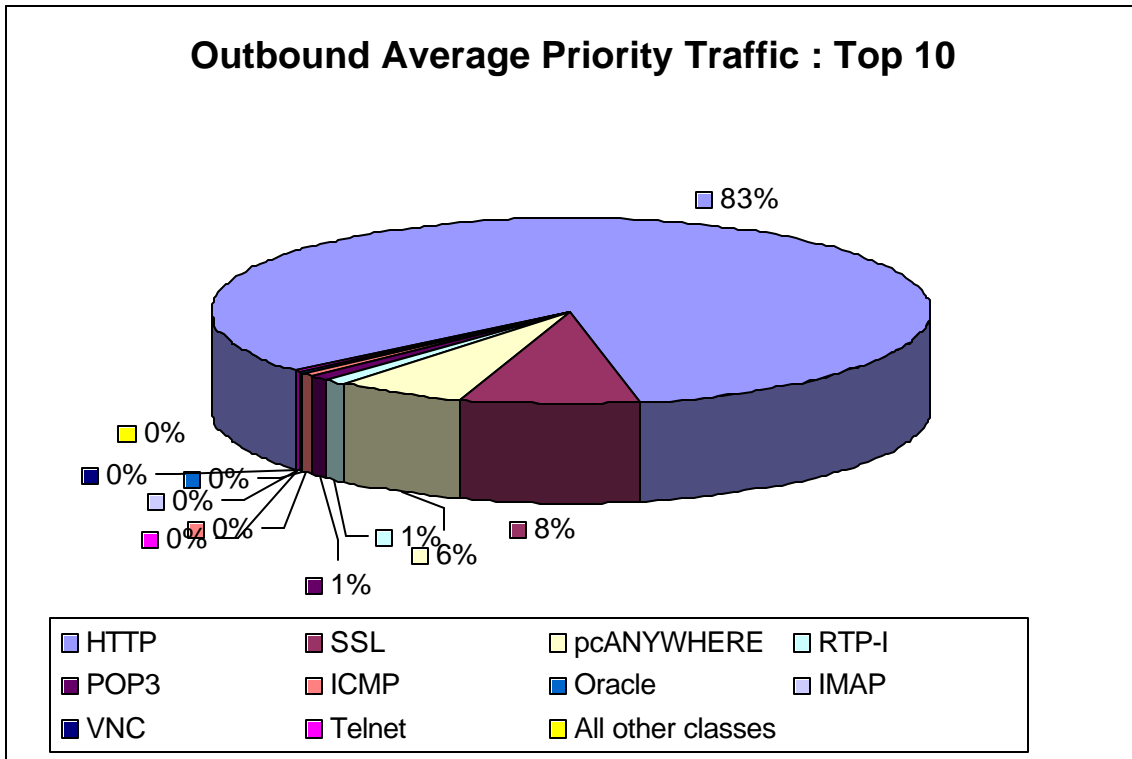
Outbound / High



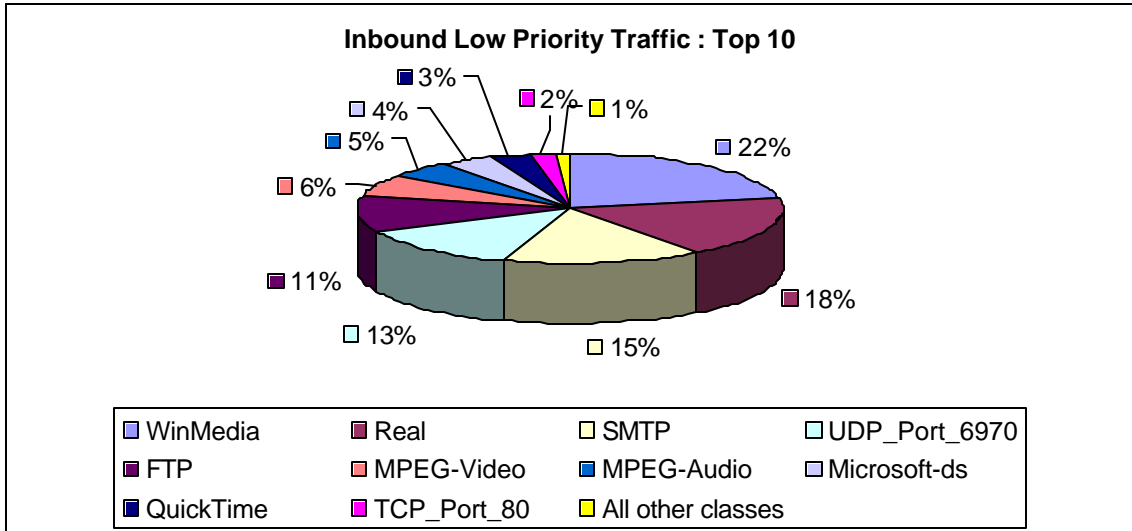
Inbound / Average



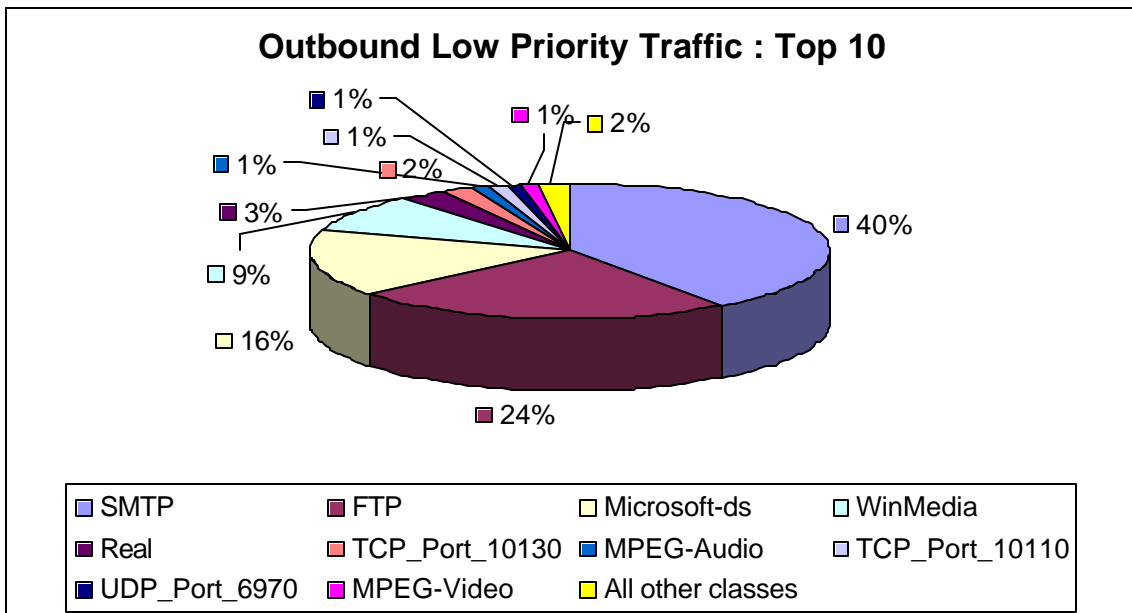
Outbound / Average



Inbound / Low



Outbound / Low



APPENDIX I

MALICIOUS DATA ON LEGITIMATE PORTS

Name	Occurrences
AddRem joke	1
Anticmos	1
Antiexe	3
AX/Frame-Exploit	1
Backdoor-RQ	1
Bonus joke	1
Downloader-W	2
EICAR	1
Exploit-MIME.gen	9
Exploit-MIME.gen.b	22495
Exploit-MIME.gen.exe	65
Flipped joke	2
Form	3
Geschenk joke	2
Happy99	4
IRC-Sdbot	1
JS/CardStealer	1
JS/Cisp	2
JS/Downloader-W	1
JS/Exploit	2
JS/Fortnight.b@M	2
JS/Fortnight@M	2
JS/IEStart.gen.c	126
JS/Kak@M	24
JS/NoClose	143
JS/Seeker.gen.e	512
JS/Seeker.gen.f	7
JS/Seeker.i	7
JS/Seeker.p	1
JS/Seeker.t	12
JS/Seeker.u	6
Linux/Exploit-SendMail	1
MessageMate joke	8
MP3Search	2
MP3Search.ldr	2
MultiDropper-AC	2
New UNIX	1
New Worm	1
One-Half.mp.3544a	1
Only joke	1
Salary joke	3
ServU-Daemon	7

Snowman joke	1
Socoten	1
Splash joke	1
StealthBoot	13
StressRelief joke	1
Suspicious IFrame.a	17
Unsafe JS	8
VBA/Generic	2
VBS/LoveLetter@MM	2
VBS/Redlof.dam	1
VBS/Redlof@M	39
W32/Aplore.htm	1
W32/BadTrans@MM	5
W32/BleBla.b@MM	2
W32/Braid.a@MM	3
W32/Bugbear@MM	118
W32/Elkern.cav.c	44
W32/Gibe.b@MM	7
W32/Gibe.gen@MM	1
W32/HLLP.Hantaner	3
W32/Hybris.gen@MM	18
W32/Klez.dam	6
W32/Klez.e@MM	1
W32/Klez.eml	18
W32/Klez.gen@MM	10
W32/Klez.h@MM	25515
W32/Klez.rar	41
W32/Korvar	4
W32/Lirva	7
W32/Lirva.a@MM	33
W32/Lirva.c@MM	10
W32/Lirva.dam	1
W32/Magistr.a@MM	89
W32/Magistr.b.dam1	2
W32/Magistr.b@MM	106
W32/Magistr.dam3	6
W32/Myparty.a@MM	12
W32/Navidad.e@M	1
W32/Nimda.eml	9
W32/Nimda.gen@MM	39
W32/Nimda.htm	5
W32/Nimda.q@MM	2
W32/Nimda@MM	61
W32/ProLin@MM	1
W32/SirCam@MM	113
W32/Ska@M	2
W32/Sobig@MM	4707
W32/SQLSpida	1

W32/Supova.worm	1
W32/Yaha.e@MM	3
W32/Yaha.g.dam	1
W32/Yaha.g@MM	1838
W32/Yaha.gen	2
W32/Yaha.k	145
W32/Yaha.k@MM	135
W32/Yaha.l	2
W32/Yaha.l@MM	7
W95/Kuang.gen	2
W97M/Assilem.c.gen	2
W97M/Class	36
W97M/ColdApe.gen	13
W97M/Ded.gen	1
W97M/Ethan.a	7
W97M/Groov.gen	2
W97M/Jerk.gen	1
W97M/Locale.gen	33
W97M/Marker.gen	87
W97M/Marker.o	3
W97M/Melissa.a@MM	21
W97M/Melissa.gen@MM	1
W97M/Nsi.e.gen	3
W97M/Proverb.gen	8
W97M/Replog.gen	2
W97M/Thus.gen	9
W97M/Titch.d.gen	88
W97M/Tristate.gen	2
WM/Cap	4
Wyx	1
X97M/Tristate.gen	2

VITA

AMIT GROVER

Personal Data: Date of Birth: January 28, 1971
 Place of Birth: New Delhi, India
 Marital Status: Single

Education: Delhi Public School, New Delhi, India; 1989
 Jawaharlal Nehru University, India; Mechanical Engineering, 1994
 IAT, India; Marine Propulsion Control Technology Course, 1995
 INS Shivaji, India; Marine Engineering Specialization course, 1996
 East Tennessee State University, Johnson City, TN, USA;
 Computer Science, Master of Science, 2003

Professional
Experience: Engineer Officer, Indian Navy, 1996-2001