Detecting Network Intrusions via a Statistical Analysis of Network Packet Characteristics

Marina Bykova, Shawn Ostermann, Brett Tjaden
School of Electrical Engineering & Computer Science
Ohio University
Athens, OH 45701
Email: {mbykova|ostermann|tjaden}@cs.ohiou.edu

Abstract—**With the growing threat of abuse of network resources, it becomes increasingly important to be able to detect malformed packets on a network and estimate the damage they can cause. Carefully constructed, certain types of packets can cause a victim host to crash while other packets may be sent only to gather necessary information about hosts and networks and can be viewed as a prelude to attack. In this paper, we collect and analyze all of the IP and TCP packets seen on a network that either violate existing standards or should not appear in modern internets. Our goal is to determine what these suspicious packets mean and evaluate what proportion of such packets can cause actual damage. Thus, we divide unusual packets obtained during our experiments into several categories depending on the severity of their consequences, including indirect consequences as a result of information gathering, and show the results. The traces analyzed were gathered at Ohio University's main Internet link, providing a massive amount of statistical data.**

Keywords— **Intrusion Detection System, suspicious activity, IP, TCP, packet analysis, packet header analysis, network monitoring.**

I. Introduction

Intrusion detection takes a greater role in the protection of a network with the growing threat of abuse of network resources. There are a number of network Intrusion Detection Systems (IDS's) that have been implemented at various research institutions and a number of commercial IDS's available (see [1] for complete listing of IDS's). Most of them, especially commercial systems like NetProwler™ [18], NetRanger™ [3] or RealSecure [7], detect well-known attacks based on their signatures.

Our goal is not to recognize all network attacks, but to determine how much information about such attacks we can obtain by looking at packet headers and not at their contents. Our approach allows us to recognize not only known attacks but also to detect suspicious activity that may be the result of a new, unknown attack.

This paper describes certain aspects of the Integrated Network-Based Ohio University Network Detective Service (INBOUNDS) [19], an IDS under development at Ohio University.

Section II provides a description of the monitored link, tools used, the types and amount of data analyzed, and the analysis performed. Section III covers the results obtained from our experiments. All detected errors are divided into categories and analyzed. Section IV summarizes our findings and also contains suggestions on improving the security of a site. Lastly, Section V provides directions for future work.

II. Description of the Experiment

A. Link Description

We monitored Ohio University's main Internet link and saw both incoming and outgoing packets. The traces were obtained on a 100Mb Fast Ethernet connection between Ohio University and its ISP and carry packets for approximately 20,000 local hosts.

B. Tools used

We used `tcpdump` [9] to capture data from Ohio University's main Internet link and `tcptrace` [11] to analyze it. A special `tcptrace` module designed for the INBOUNDS project [19] was used to send data to other modules of the IDS and report abnormal behavior.

C. Packet Analysis

Analysis of the monitored link shows that almost all packets on the link are IP packets and the great majority of those are TCP packets. UDP traffic comprises approximately 2 percent of the monitored traffic, and IP packets that are neither TCP nor UDP make up an even smaller portion of the total data. Thus, our analysis is based on the IP and TCP headers of packets from the monitored traffic.

Descriptions of some IDS's list the types of unusual packets they captured during continuous operation [12] and others give descriptions of attacks based on individual malicious packets that can cause harm [17], [5]. We analyzed many types of packets outlined in prior research that might cause damage, applied this information to the IP and TCP header fields, and will summarize it in this section.

C.1 IP Header Analysis

1. **Packet Size**. The IP header length should always be greater than or equal to the minimal Internet header length (20 octets) and a packet's total length should always be greater than its header length [13]. If any of these statements do not hold for a given packet, it is invalid and should be discarded at the destination host.

IP packets that carry transport layer protocols known to the system, currently only TCP, are checked to be large enough to hold the entire header of the next layer protocol.

0-7803-6661-1/01/$10.00 ©2001 IEEE

2. **'Time to Live' Field**. The 'Time to Live' (TTL) field can be used by an attacker to explore the topology of a remote network [5]. When attempting to map the topology, a combination of `traceroute` [8] attempts can provide a good picture of the network. In most cases, however, it is impossible to determine for what reason `traceroute` was used.

Low TTL values can also be used in subtle attacks that try to subvert a monitor. Bro [12] has a detailed description of such attacks in which an attacker sends packets with low TTL values and retransmits the same packets with different data and a larger TTL so that only the retransmitted packets will reach the destination host.

We record packets that have small TTL values, but an important part of our analysis is to determine why such low values were used. For instance, limited broadcast packets should always keep a small TTL value and thus should be excluded from the list of suspicious packets so as not to generate alarms.
3. **IP Address**. The IP address field is unprotected from spoofing, i.e. substituting it with an IP address that does not belong to the sender, and the source address extracted from a single packet can not be easily verified. Source address spoofing becomes harder with protocols that maintain a connection and have state, such as TCP, but it is still possible with, e.g. source routing.
The problem of determining the validity of source addresses can not be easily solved when access to a network is unrestricted and a monitor sees both incoming and outgoing traffic. However, a number of addresses that are certainly invalid can still be identified. Prior literature contains examples of network attacks that use the same source and destination IP addresses, the so called "land attack" described in [17], [6].

Another category of invalid addresses is private internet addresses [15]. Private addresses are invalid in public domains and should be filtered out by routers connected to private networks. However, our experience shows that a number of packets containing private addresses do exist in the public domain. Empirical results from Bro [12] confirm this fact as well.

There are certain special cases of IP addresses [16] that can not be used as either source (broadcast addresses), destination ("this network" addresses), or either kind (loopback addresses) of address on a public internet. Many of them are based on the definition of "network number" and "subnet number". The difficulty in detecting these types of internet addresses arises from the variable length of network prefixes, and in general we do not know the network prefix length for any given IP address. However, in our analysis we do look for the special cases of IP addresses and record packets that clearly belong to one of the special cases.
4. **IP Options**. From all available IP options [13], we look only for the strict source routing option because it is to be used only for debugging purposes and should not appear in modern internets. There are a number of attacks (see [6] for description) that use strict source routing together with a spoofed source IP address to be able to receive responses and establish bogus communication with the target host. Other IP options, at the time of this writing, are not known to harm the destination.
5. **Overlapping Data**. Overlapping fragments in which the two fragments do not agree on the contents of the overlapped region and retransmitted packets that carry different data always violate protocol specifications and should generate alarms. Several IDS implementations ([12], [17]) report such cases, but we currenly do not look at the contents of monitored traffic. This remains for future study.

C.2 TCP Header Analysis

1. **Packet Size**. The data portion of the first IP packet of a fragment set containing a TCP packet should be large enough to hold an entire TCP header. If a TCP header includes many long options, then some of them may be truncated and carried in the next IP packet. However, the required part of TCP header (20 octets) is normally present entirely in one IP datagram. Splitting TCP headers is sometimes used to pierce firewalls, and that is the reason why we check for very short packets.
2. **Port Numbers**. Neither the source nor destination TCP port number can be zero [14]. We record all packets where either one of these two port numbers is equal to zero. Also, source and destination port numbers usually differ. There is no rule that they cannot be equal but such cases are not common and we record them as suspicious.
3. **TCP Flags**. According to the TCP standard [14], URG and PSH flags can be used only when a packet carries data. Thus, for instance, combinations of SYN and URG or SYN and PSH become invalid. Moreover, any combination of more than one of SYN, RST, and FIN flags is also invalid [1]. We check whether a TCP packet has a valid combination of flags and any protocol violations are reported by the system.
4. **Acknowledgements for never-sent data**. Bro [12] is known to report such cases but we currently do not have statistics for such violations and do not include this in the analysis. This capability will be added in the future.

D. Analyzed Data

During our experiments we analyzed traces gathered May 2000 through November 2000 at different times of the day on Ohio University's main internet link which carries data for approximately 20,000 hosts that reside at the university. Each trace file consisted of several million packets and the total number of analyzed packets was over 330,000,000. We have seen over 6,600,000 complete or partial TCP connections. The total number of reported warnings over all of the analyzed data was approximately 250,000.

III. RESULTS

This section provides a detailed analysis of obtained results and description of all types of generated errors. All errors pro-

[1] According to the T/TCP RFC [2], a packet that includes both SYN and FIN flags might be valid if it carries a CC or CC.NEW option. In our analysis, we take into account these options even though the implementation of T/TCP is experimental and is not a current standard.

TABLE I
DETECTED ERRORS

Type	Packets	Error %	Total %
Packets with low TTL values	141046	55.54%	0.0421%
Packets with the same port numbers	44556	17.55%	0.0133%
Packets containing private IP addresses	22264	8.77%	0.0066%
Packets with IP address violations	722	0.28%	0.0002%
Packets with invalid TCP flags	288	0.11%	0.0001%
Packets containing zero port number	206	0.08%	0.0001%
Packets with strict source routing option	0	0.00%	0.0000%
Too short packets	0	0.00%	0.0000%
Total number of errors	253938	100.00%	0.0758%

TABLE II
DISTRIBUTION OF PACKETS CONTAINING PRIVATE IP ADDRESSES

Private IP Address Range	From	To	Total
Class A private IP addresses (10.0.0.0/0.255.255.255)	6524	5825	9680
Class B private IP addresses (172.16.0.0/0.15.255.255)	29	5443	5472
Class C private IP addresses (192.168.0.0/0.0.255.255)	2077	5654	7576
Total number of packets	8630	16931	22264

duced by the system are summarized in Table I. It can be seen that the system did not generate all recognized types of errors and did not see all types of known violations, which tells us that either the amount of analyzed data was not large enough to detect such packets and calculate their rate or they do not exist in numbers on the Internet.

A. Private IP Addresses

Currently, Ohio University utilizes a few private networks that use the class A private IP addresses (10.0.0.0/8) and are protected with firewalls that perform IP address translation (NAT). Ohio University does not use the other ranges of private addresses and has not done so during recent years. Thus, packets destined to private IP addresses other than these class A addresses could not be caused by old configurations left from previous address schema.

Results obtained during our experiments showed a large number of TCP packets sent either to or from private IP addresses. Moreover, the logged packets contain IP addresses that belong to all classes of private networks. The distribution of these packets containing private IP addresses by address ranges and the type of address being private — source, destination, or both — is shown in Table II. Note that the total number of packets that fall into each address range is not necessarily equal to the sum of packets going to and from private addresses from that address range because some of the packets had both the source and destination addresses in private address space.

We found that TCP packets destined to private IP addresses are sent by various Ohio University hosts that run different operating systems and have different configurations. Thus, the presence of such packets can not be explained either by errors in implementation or by improper default configuration of a certain operating system and must have a different origin. Such packets do not belong to the Ohio University IP address space and thus tend to leave the domain. They are normally blocked or discarded either due to absence of routes for such IP addresses or after the maximum number of hops is reached.

Packets with private source addresses can be divided into several categories. Some such packets come from hosts with private IP addresses assigned to them and are seen by the monitor due to errors in router software or configuration. The other group of such packets have spoofed source addresses. Unfortunately, it is impossible to track back to any of the hosts that sent such packets using their hardware addresses because all packets were gathered after they went through the router and thus have the same MAC address.

B. Other IP Address Violations

This subsection describes all other IP address violations that were detected during our experiments. These kinds of violations come from so-called "special" IP addresses [16] that may not be legimately used as the source address, destination address, or either one.

In general, IP addresses can be represented using the following notation:

$$\textit{IP-address} = \{<\textit{network\_number}>, <\textit{host\_number}>\}$$
or
$$\textit{IP-address} = \{<\textit{network\_number}>, <\textit{subnet\_number}>, <\textit{host\_number}>\}$$

However, we analyze IP addresses from the global internet and, as a rule, can not know the network or subnet number lengths or even whether a particular network has subnets. Thus, we use the first notation from the two given above and do not include subnet numbers in our analysis. We also use the notation *"1...1"* to indicate that a field contains all 1 bits.

Using the assumptions above, some common special cases of IP addresses are as follows:
1. $\{0, 0\}$ This host on this network. Can only be used as a source address.
2. $\{0, <\textit{host\_number}>\}$ Specified host on this network. Can only be used as a source address.
3. $\{1...1, 1...1\}$ Limited broadcast. Can only be used as a destination address, and a datagram with this address must never be forwarded outside the network of the source.
4. $\{<\textit{network\_number}>, 1...1\}$ Directed broadcast to specified network. Can only be used as a destination address.
5. $\{127, \textit{any}\}$ Internal host loopback address. Should never appear outside a host.

Table III summarizes all such errors detected by the system and provides the total number and percentage of packets that fall in each category. As it can be seen, the system has not recorded any packets of types 1 or 4. All other types, however, were present in the trace files.

Case 2 shows packets that were sent to network 0 and thus did not have a potential destination. The great majority of such packets were SYN packets sent to well known TCP port numbers. They received no responses because no host can have a

TABLE III
DETECTED IP ADDRESS VIOLATIONS

Case	Description	Used As	Packets	Percent
1	This host on this network	destination	0	0.0%
2	Specified host on this network	destination	197	27.3%
3	Limited broadcast	source	523	72.4%
4	Directed broadcast to a network	source	0	0.0%
5	Internal host loopback address	either	2	0.3%
	Total number of packets		722	100.0%

zero network number in the global internet. Other packets that fell in this category were UDP packets that appeared to belong to name service traffic. We believe that both such types of erroneous packets were caused by misconfigured software. This kind of packet belongs to outgoing traffic and is neither dangerous nor useful. We recommend they be blocked by routers.

Case 3 provides statistics for packets sent from the IP address 255.255.255.255. A portion of such packets were ICMP 'UDP PORT UNREACHABLE' packets. The packets included error messages for different port numbers, but the majority of them were for port 2519. We could not know the MAC addresses of the machines that sent these packets and thus could not determine what type of hosts generated the packets. Being invalid, the packets should not be present on a network.

Other packets coming from the limited broadcast address inadvertently allowed us to detect a number of large network scans. During these scans, a SYN packet was sent to a particular port, usually 23 or 111, on every host on a network including 0. We have discovered that some SYN packets sent to network addresses — those with the last octet equal to 0 — get replies back from the IP address 255.255.255.255. These replies could be either RST or SYN packets, but in either case they were sent from the same limited broadcast IP address. These replies were invalid and could be caused by errors in the software installed on a host or router on the target network.

Case 5 shows that some packets which should never leave a particular host appear on the network. In our case, both the source and destination addresses were loopback addresses and most likely left the host due to an erroneous implementaion of TCP/IP. However, it is possible to spoof the source address and use the loopback addresses as a source address. Therefore, it is recommended that all packets containing internal loopback addresses are blocked at the router.

C. Low TTL Values

We know that packets with small TTL values can be a precursor to or a part of a network attack, but they can also occur for legitimate reasons. A number of packets with low TTL values can be caused by routing loops, and such cases are relatively easy to recognize. Other packets could be a result of `traceroute` usage. In general, it is very difficult to determine for what purpose `traceroute` was used, and in order to make any conclusions additional information is needed. This should be an area of future study.

TABLE IV
TYPES OF PACKETS WITH ZERO PORTS

Case	Category	Packets	Percent	Possible Cause
1	Malformed packets in the middle of a connection	46	22.3%	poor implementation
2	Invalid SYN packets	8	3.9%	poor implementation
3	ACK packets from port 0 to port 6	109	52.9%	possible attack
4	RST packets from port 0 to high port numbers	38	18.4%	port scan
5	Other	5	2.4%	varies
	Total number of packets	206	100.0%	

D. TCP Packets with Zero Ports

Another category of errors generated by the system is TCP packets where at least one of the port numbers, source or destination, has a value of zero. The large majority of these packets followed certain patterns and were easily divided into several categories. Table IV shows all the categories and the percentage of packets that fall into each category.

A large portion of the packets of this type is formed by invalid TCP packets. Case 1 provides numbers for malformed packets that could correspond to open connections. Such packets were issued in the middle of a connection, were not valid, and likely were ignored at the destination host. In the majority of such cases, the source port number was set to zero and the real source port number appears to have been used as the destination port. Many such packets had invalid TCP flags or TCP header length and did not carry valid data. The rate of this kind of packets was not high, but the pattern is obvious and could have been caused by TCP implementation errors.

Case 2 also shows erroneous implementation or incorrect usage of a network application. These packets were sent to a zero port in order to open a connection. Packet retransmissions followed the standard time increments and do not look dangerous.

The next group of packets with zero port numbers (Case 3 in Table IV) looks more suspicious than the packets described above, and all of the packets from this group followed a very specific pattern. The packets were simple ACK packets where source and destination ports were 0 and 6 respectively and all them advertised a TCP window of size 0. Over 8% of these packets had private IP addresses as the source IP address. It is our opinion that they were sent on purpose using the same tool. The number of such packets was rather large — they comprise over 50% of all packets with zero ports — and could be a part of a network attack or host detection.

The next category of packets (Case 4 in Table IV) also followed a specific pattern and could have been generated using one tool. Most of the packets from this category were sent from one specific IP address and all were sent from port 0 to ports 1024 or 3072. All of them were RST packets that acknowledged data and advertised a TCP window of size 0. They were sent to many different Ohio University hosts, one or two packets per host, and were most likely port scans.

TABLE V
DISTRIBUTION OF PACKETS CONTAINING SAME SOURCE AND DESTINATION TCP PORTS

Type	Connections	Packets	Percent
Port 53	68	268	0.60%
Other low TCP ports	2	14	0.03%
High TCP ports	74	44261	99.37%
Total number of packets	144	44543	100.00%

TABLE VI
TYPES OF PACKETS WITH INVALID TCP FLAGS

Type	Packets	Percent
Corrupted packets	240	83.3%
FIN and RST to close a connection	28	9.7%
FIN and RST by itself	16	5.6%
PSH set in second SYN	4	1.4%
Total number of packets	288	100.0%

All other packets were united under the 'Other' category. Table IV tells us that there were only 5 such packets. 2 of them were RST packets sent in response to bogus SYN packets with zero destination port. These packets follow internet standards [14] and should not harm anything.

The IP addresses of other packets from this group did not belong to pairs of communicating hosts. However, they look like legimate packets that were either corrupted at the sending end or during transmission.

E. Same Source and Destination TCP Ports

The next category of erroneous packets that we analyze is TCP packets that used the same number for both source and destination TCP ports. Table V summarizes such packets, divided into groups based on their port numbers. Packets sent to and from port 53 — which is assigned to DNS traffic — were separated into their own group because they comprise a disproportionately large number of such packets.

Analyzing the packets, we found that all packets that fall into the first group, i.e. those that were sent to and from port 53, were SYN packets sent from external hosts to Ohio University's main DNS servers and RST packets sent back from our DNS servers to the external hosts. Presence of such packets can be easily explained because the Ohio University main DNS servers do not use TCP to exchange domain name information but use UDP. All attemps to connect to the servers on TCP port 53 were rejected. We view this traffic as legitimate.

The next group of packets had communication on low-numbered TCP ports (less than 1024) other than 53. As can be seen from Table V, there are only 2 such cases. The first case was an attempt to connect from port 20 to port 20 which was not successful. The other case was a short connection from port 80 to port 80 which appeared to us to be a legitimate connection.

The majority of all packets with the same TCP port numbers on each end were high ports. The number of packets that belonged to each particular case differs from several packets to several thousand packets. Most of them looked like regular connections that simply happened to run on the same ports.

Of course, there are a number of cases when packets that did not seem to belong to existing connections were issued on the same port numbers, for instance single FIN packets or ACK packets not preceeded by a proper connection establishment. However, we believe that the likelihood of sending such packets on the same port numbers is not higher than the likelihood of sending similar packets on other combinations of TCP port numbers based on the collected data. We conclude that presence of matching port numbers in a single TCP packet is not a good criteria for determining whether a packet is dangerous.

F. Invalid TCP Flags

The last category of erroneous packets that we include in our study is TCP packets with invalid flags. Table VI lists the different types of such packets that we saw in the trace files.

We named the largest group of packets that carried invalid TCP flags corrupted packets. This group includes different packets with various combinations of TCP flags where the source and destination IP addresses belonged to hosts that were communicating at the time of the traces. We believe these packets were either malformed at the sending end or corrupted during transmission and thus should have been ignored by the communicating machines. Unfortunately, we could not verify their checksums because the data portion of the packets was not available in our analysis. The corrupted packets include:

1. Packets with IP addresses and TCP port numbers belonging to open connections but carrying invalid data.
2. Packets in which the source port number was used as the destination port, and the source port had a random value. A large portion of such packets had the source port number set to zero. This same error was described during our analysis of TCP packets containing zero port numbers (subsection D).
3. Packets with correct values for the source port number, i.e. corresponding to an open connecition, but incorrect values for the destination port number.
4. Packets with IP addresses belonging to communicating hosts but neither one of the source or destination port numbers belonged to connections that were open between the hosts.

Even though all these packets carry valid IP addresses, the first type of the corrupted packets is the most innocent because the packets might interrupt at most one connection between the hosts. Other errors introduce new port numbers and thus might be more harmful.

The next type of packets with invalid TCP flags were packets that had both FIN and RST flags set and were sent to close a connection. Such packets were sent:

1. In response to first SYN to reset the connection as RST packets;
2. As the second FIN after the first FIN packet went in the other direction;
3. After both FIN packets when the connection is almost closed;

4. After a RST packet in the same direction.

There is no one single explanation of all such packets. A number of them, especially packets sent to close longer connections, were most likely legitimate packets. However, we believe that such packets should still be blocked because they violate existing standards.

Other packets that had both FIN and RST flags set were sent on their own without any other communication between the hosts. Since packets from this group do not have connections associated with them, the packets could be a part of a host detection or fingerprinting attack.

Table VI also shows that there were a several SYN packets which were sent with the PSH flag set. Even though a TCP packet can have a PSH flag set only when it carries data [14], these packets were apparently accepted during connection establishment and the communication continued. Normally, such packets can not cause damage but nevertheless they should not be accepted as valid packets at their destination.

IV. Conclusions and Recommendations

In many cases, it should be easy to secure a site at the router against possible attacks that use invalid values of IP or TCP header fields through proper router configuration. The majority of router software should allow to filter out at least the following packets:

1. Packets carrying private IP addresses where either the source or destination address is private;
2. Packets with other IP address violations described in Section III subsection B;
3. Packets with zero port numbers;
4. Source routed packets.

These simple filters will noticeably reduce the number of warnings generated by the IDS. Other suspicious cases can be analyzed and handled by the system itself. For instance, INBOUNDS is capable of determining whether a TCP packet belongs to an open connection or not. Based on its decision, some packets with invalid TCP flags may be blocked while others may be allowed to go through.

Summarizing the results, we should say that we do not consider a large portion of the packets that generated warnings harmful and believe they could be caused by poor IP or TCP implementations or other similar errors. On the other hand, we were able to catch a number of cases where erroneous packets could not belong to legitimate traffic. Such cases can be analyzed so that knowledge obtained about them can be integrated into the system to make intrusion detection more efficient.

V. Future work

As previously mentioned, we did not include the contents of packets in our analysis and considered only packet headers. In the future, we intend to expand the analysis to cover the data itself. Examples of future directions include comparison of original and retransmitted packets for cases when such packets do not agree on data contents. This is often done to subvert monitors for the purpose of network attacks. We would also like to detect fragments that carry overlapping data such that the contents of the overlapping region are different in the different fragments.

In our research, we did not attempt to detect TCP packets carrying acknowledgements for data that have not been sent. Such packets are not hard to detect and would be interesting to analyze. We believe that this bears future research. We also leave more detailed analysis of packets with low TTL values for future study.

Lastly, this work can be expanded to include analysis of data from different networks with more diverse traffic.

Acknowledgements

The authors are indebted to Ethan Blanton for reviewing this document and many useful suggestions and corrections. The authors also acknowledge the Ohio University Communication Network Services staff, in particular Theresa Kelleher and Todd Acheson, for providing us with the data and necessary information that made our research possible.

References

[1] J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner, "State of the Practice of Intrusion Detection Technologies", http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html, Jan. 2000.
[2] R. Braden, "T/TCP — TCP Extension for Transactions Functional Specification", RFC 1644, Jul. 1994.
[3] Cisco, The NetRanger Intrusion Detection System, http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/index.shtml, 2000.
[4] Fyodor, "Remote OS detection via TCP/IP Stack FingerPrinting", http://www.insecure.org/nmap/nmap-fingerprinting-article.html, Oct. 1998.
[5] R. Gula, "How to Handle and Identify Network Probes", http://www.network-defense.com/papers/probes.txt, Apr. 1999.
[6] Internet Security Systems, Real Secure Attack Signatures, http://documents.iss.net/literature/RealSecure/Signatures_5.0.pdf, 2000.
[7] Internet Security Systems, Real Secure, http://documents.iss.net/literature/RealSecure/rs_guide.pdf, 1999.
[8] V. Jacobson, `traceroute`, ftp://ftp.ee.lbl.gov, 1989.
[9] V. Jacobson, C. Leres, and S. DMcCanne, `tcpdump`, http://www.tcpdump.org, Jun. 1989.
[10] S. McCanne, C. Leres, and V. Jacobson, `libpcap`, http://www.tcpdump.org, Jun. 1994.
[11] S. Ostermann, `tcptrace`, http://www.tcptrace.org, 1994.
[12] V. Paxson, "Bro: A System for Detection Network Intruders in Real-Time", *Computer Networks*, 31(23-24), pp. 2435-2463, 14 Dec. 1999.
[13] J. Postel, "Internet Protocol", RFC 791, Sep. 1981.
[14] J. Postel, "Transmission Control Protocol", RFC 793, Sep. 1981.
[15] Y. Rekhter, B. Moskowitz, D. Karrenberd, G. J. de Groot, and E. Lear, "Address Allocation for Private Internets", RFC 1918, Feb. 1996.
[16] J. Reynolds and J. Postel, "Assigned Numbers", RFC 1700, Oct. 1994.
[17] R. Sekar, Y. Guang, S. Verma, and T. Shanbhag, "A High-Performance Network Intrusion Detection System", in *Proceedings of the 6th ACM conference on Computer and communications security*, pp. 8-17, 1999.
[18] Symantec, NetProwler, http://enterprisesecurity.symantec.com/products, 2000.
[19] B. Tjaden, L. Welch, S. Ostermann, D. Chelberg, R. Balupari, M. Bykova, M. Delaney, A. Mitchell, S. Li, D. Lissitsyn, and L. Tong, "INBOUNDS: The Integrated NetworkBased Ohio University Network Detective Service", *4th World Multiconference on Systemics, Cybernetics, and Informatics (SCI'2000)*, Jul. 2000.