

# Statistical Opportunities in Network Security

David J. Marchette\*

Keywords: Network Monitoring, Denial of Service Attack, Passive Fingerprinting, User Profiling, Computer Security

## Abstract

We discuss several problems that arise in network monitoring and computer security that lend themselves to statistical analysis. We briefly describe the problems and give some preliminary results.

## 1 Introduction to Network Data

The data we will consider in this paper are IP packets. These can be collected off a network interface through software called a “sniffer” or network sensor (in this case the program `tcpdump`). A packet consists of a header, which contains routing and protocol information, and data, which contains the information being transferred over the network. We will consider only the header information. In addition to the header information, a network sensor also places a time stamp on the packet, so we also know the time at which the packet arrived at the sensor. We will refer to the network monitored by the sensor as the protected network.

Network traffic is typically very high volume. To give an example, a network of around 10,000 machines was monitored for one hour on January 31, 2002, from noon to 1pm. For the most part, all machines on the network are single user machines, and so there are typically no more than a thousand or so users on the network at any one time. During this hour there were over 7 million IP packets, resulting from over 150,000 individual sessions.

A network session (such as an email message, web browsing, telnet, ftp, etc.) consists of many packets. The information transmitted is broken into discrete chunks, which are then sent out to be routed to the destination, where they are put back together into a coherent whole. The packets are not guaranteed to arrive in order, so mechanisms are in place to ensure that they can be correctly reassembled. Also, there are mechanisms (in TCP, the protocol we will be considering here) to detect lost or damaged packets, and request that the packet be resent. These mechanisms are controlled by the information in the header.

The packet header consists of two parts: the IP header, which contains routing information, and the TCP header which contains protocol information. These headers are illustrated in Figures 1 and 2. More information on these can be found in Stevens [1994] and Marchette [2001], or in any book on TCP/IP. The most important fields, for the purpose of routing the packets, are the source and destination IP addresses, which determine the machines involved in the transaction. These are 32 bit numbers, usually written in “dotted” format, such as: 10.10.45.123.

The TCP header contains source and destination ports, which can be thought of as an additional 2 bytes of addressing. This provides a mechanism to uniquely identify the session, allowing the computers to have multiple simultaneous sessions. Often, these ports determine the application (such as web or email) corresponding to the session.

---

\*Naval Surface Warfare Center, Code B10, 17320 Dahlgren Rd., Dahlgren, VA, 22448; marchettedj@nswc.navy.mil, 540-653-2736, 540-653-2641 (fax)

Version	Length	Type of Service	Total Packet Length	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source IP Address				
Destination IP Address				
Options (if any)				

Figure 1: The IP header. Each row corresponds to 4 bytes. The header the consists of these bytes in order from left to right, top to bottom. The fields are all either 1, 2 or 4 bytes, with the exception of the flags and fragment offset fields, which are 3 bits and 13 bits respectively.

Source Port			Destination Port	
Sequence Number				
Acknowledgment Number				
Length	Reserved	Flags	Window Size	
Checksum			Urgent Pointer	
Options (if any)				

Figure 2: The TCP header.

The flags are used to initiate and control the session. There are six flags, denoted S (SYN or synchronize), R (RST or reset), A (ACK or acknowledge), F (FIN or finish), U (URG or urgent) and P (PSH or push). Again, details are available in any book on TCP/IP. For our purposes, it is enough to know the basics of how a TCP session looks. When a user browses a web page, a session is initiated by the client computer sending a packet with only the SYN flag set (called a “SYN packet”). The server, if it accepts the connection, sends an acknowledgment (a packet with both the SYN and ACK flag set: a SYN/ACK packet). Finally, the client sends an ACK packet, and the session is initialized. In addition to these flags, sequence and acknowledgment numbers are used to provide ordering of the packets and insure that any missing packets are detected and resent. Packets now may flow in either direction, using ACK packets to acknowledge receipt, and PSH packets to indicate that the data should be “pushed” up to the application. The session is closed off with FIN packets, which are acknowledged, or RST packets, which immediately close the session.

There are a number of ways to attack a computer, and the ultimate goal of the attack determines the appropriate method. One may simply be after information about the network or computer, such as what IP addresses are populated and what services are available. This is generally a prelude to a subsequent attack on the network or computer. One may wish to deny legitimate users access to the computer. Or one may want user (or super user) level access to the computer. We will discuss several ways in which statistics can play a part in the analysis of these attacks. In Section 2 we will look at attacks that deny access to the computer by legitimate users. In Section 3 we will look at ways to determine the operating system that a computer (such as an attacking computer) is running. In Section 4 we will discuss some methods that can be used to gather information about users, for the purpose of determining when users change their profile. This might be of use in detecting masqueraders.

## 2 Denial of Service Attacks

The purpose of a denial of service attack is to deny legitimate users access to a service. There are many ways to do this, depending on the level of access that the attacker can gain on the server. We will discuss only one of these.

Recall that a session is initiated through the “three-way handshake” in which an initiating SYN packet is acknowledged with a SYN/ACK, which is subsequently acknowledged. This is the key to an attack called the “SYN flood”. The idea is to initiate a large number of sessions, without completing the handshake. The server fills its session queue with these bogus sessions, and legitimate users are locked out. The attacker utilizes two clever ideas to make this attack most effective. First, they often utilize a large number of attacking machines, maximizing the number of packets received by the server. Second, and most importantly from our perspective, they “spoof” their IP address. That is, they put a random IP address in the header as the source IP, thus making it impossible for the server to know who the attacker is, or to block an attack by denying access to the attacker’s IP.

The result of this spoofing is that the SYN/ACK sent by the server is received not by the attacker but by a random IP address out on the Internet. Thus, our network sensor receives unsolicited SYN/ACK packets from the victim computer, and we can use these to make estimates about the number of victims and the size of the attacks. We can do this without any requirement of the victim to report the attack (or even detect it), and without adding any burden to the network (such as additional packets sent to the victim).

Thus, the sensor observes a subset of the responses to the attack. We will assume that the spoofed addresses are randomly selected (perusal of several attack codes indicates that this is often the case), and that all unsolicited SYN/ACK packets are the result of an attack on the source IP (the victim). This last assumption is the most questionable, as there are a number of reasons one might receive such a packet, such as a stealthy scan of the protected network.

We will restrict ourselves in this paper to attacks against web servers. Figure 3 depicts an attack on a single victim. The streaking evident in the figure is the result of resends: when the victim does not receive a response to its SYN/ACK packet, it sends another, and keeps resending, with increasing wait times, until a time-out period has been reached, at which point it determines that the session is not legitimate and closes it off. This results in multiple (essentially identical) packets from a single attack packet. From here on we will filter out all resends, and concentrate only on the first packet received.

Figure 4 depicts a typical attack. The spoofed IP addresses appear to have been selected randomly, uniformly, and independently (all of these claims are subject to statistical tests).

Figure 5 depicts another victim. This illustrates both the non-stationarity of some attack processes, and structure indicating that the spoofed IP addresses are not chosen randomly. This may be evidence of a scan of the protected network, or of a different type of attack tool.

Treating an individual attack as a sequence of packets received from a single victim with no inter packet arrival time exceeding 5 minutes, we can determine the number of attacks that we are observing as a function of time. This is depicted in Figure 6. There are a number of times the sensor was off-line, and these are indicated by the regions of zero attacks. Note that since we are counting attacks against individuals, attacks against several servers at the same organization are counted as distinct attacks.

Note that something happened in mid September to dramatically increase the number of attacks. It is not known whether this is the result of September 11, or if this was the release date of a new attack tool, or if there was some other reason for the change. More information on this study can be found in Marchette [2002].

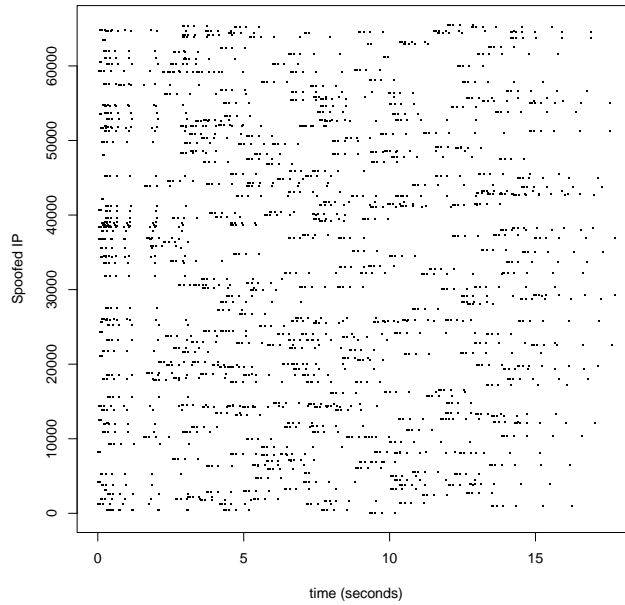


Figure 3: A victim. The horizontal axis corresponds to time and the vertical to destination IP address. These are the addresses spoofed by the attacker.

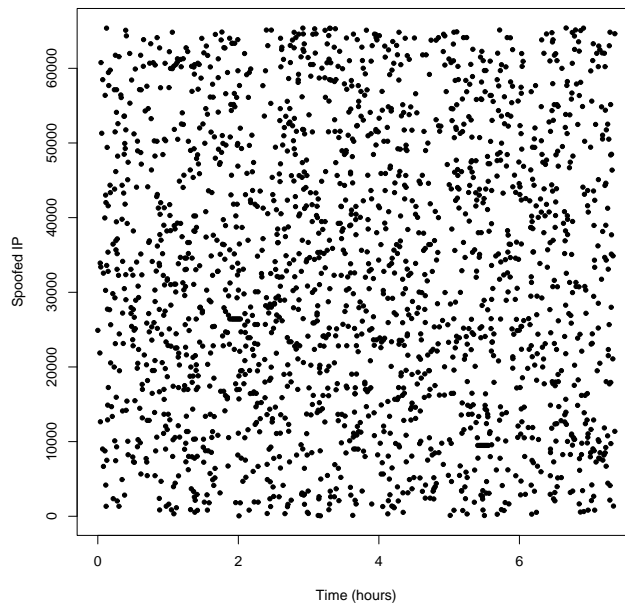


Figure 4: Another victim. The horizontal axis corresponds to time and the vertical to destination IP address. These are the addresses spoofed by the attacker.

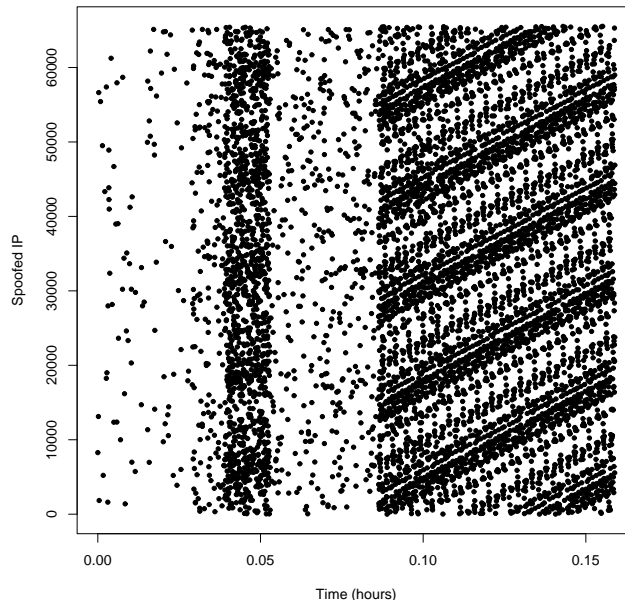


Figure 5: Another victim. The horizontal axis corresponds to time and the vertical to destination IP address. These are the addresses spoofed by the attacker.

### 3 Passive Fingerprinting

Fingerprinting is the assignment of an operating system (or operating system and version number) to an IP address from observing network traffic or responses. Active fingerprinting works by sending carefully crafted packets to the IP address and observing the response. Since the protocol does not specify the required response to packets that are not a normal result of network traffic (such as a packet with all the flags set), the operating system is free to respond in a number of different ways. Different operating systems have been written to respond differently, and this can be used to determine the remote computer's operating system. Determining the operating system of a computer can be useful in crafting an attack, or in determining the likely attacks (or attack tools) that it might employ. It can also be used to detect a spoofed IP address, if one has a database of operating systems to match against, and similarly to detect crafted packets, which are likely to be attacks.

Passive fingerprinting relies on monitoring normal network traffic, and determining the operating system of the remote machine by observing the packets it sends in the normal course of its sessions. This relies on a number of options in the header that the operating system is free to set. We will look at a brief study of passive operating system fingerprinting. The data for this study is available in the files `ptrain.txt` and `ptest.txt` (the training and test data, respectively).

An example of a field value that the operating system can select is the time to live. This is an 8-bit number that is decremented at each router that the packet traverses. If a router decrements the value to zero, it drops the packet, and sends back an ICMP message indicating that the packet's time has been exceeded. This is to prevent a lost packet from bouncing around the Internet forever. Operating systems may set this to any value they wish, although it is suggested that they choose a value larger than any likely path length.

We collected data on 3806 machines over a period of about 6 months. For the sessions these machines initiated, various features such as the average TTL, average type-of-service, window size, etc. were collected. These are in the data files `ptrain.txt` and `ptest.txt`. The operating systems for these IPs were determined via an accreditation database. The problem is to determine which operating system a machine is running

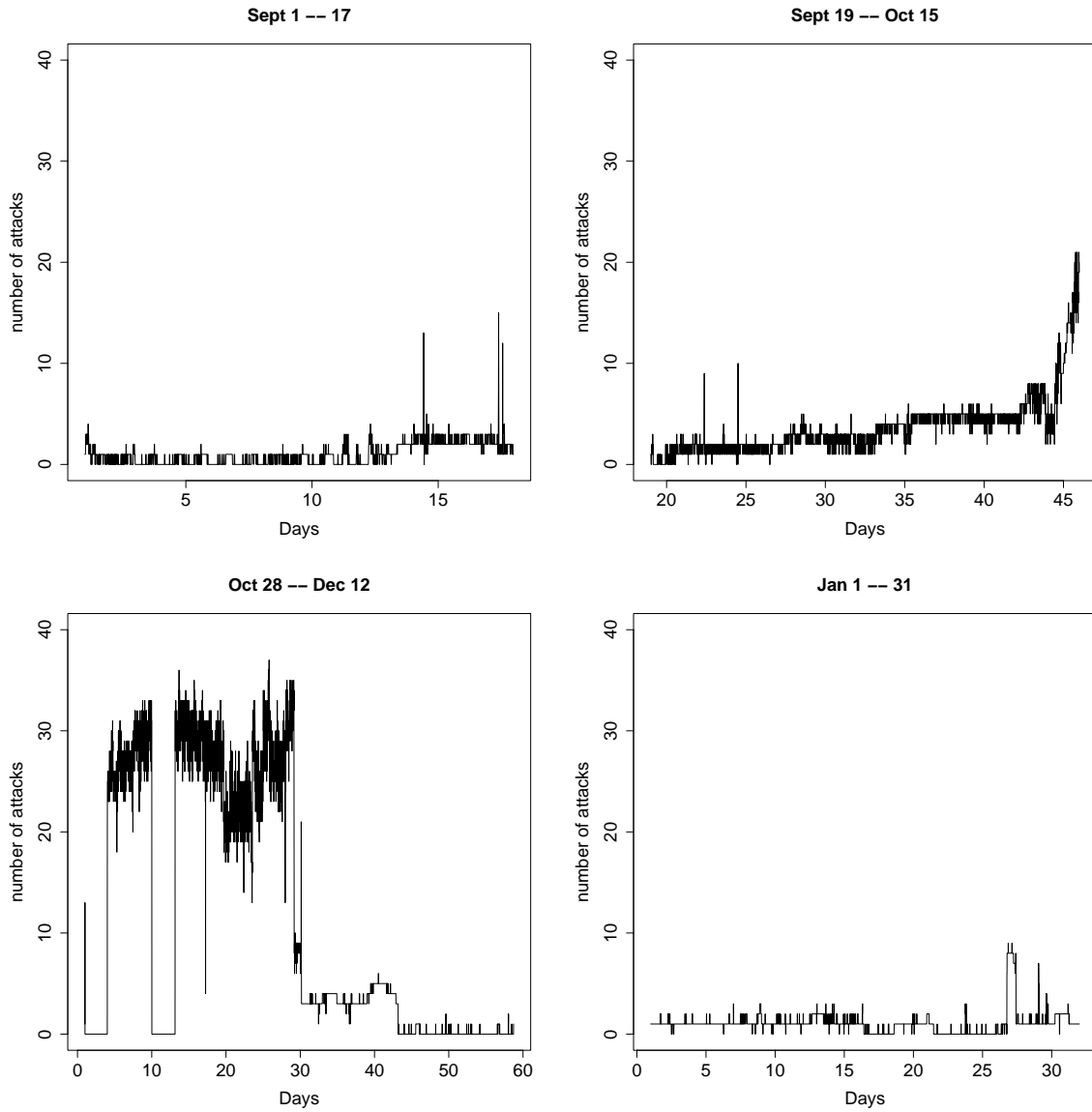


Figure 6: Number of attacks for four distinct time periods. The attacks are defined as a collection of packets from a distinct victim with no inter packet arrival time exceeding 5 minutes.

from these statistics. We used a 3-nearest neighbor classifier, to illustrate the results that one can obtain. Figure 7 shows a parallel coordinates plot for some of the data. This indicates that there is some separation in some of the fields. Table 1 shows the results of the classifier built on the training data and tested on the test data. Calling all operating systems “windows” results in an error of 0.074, while this classifier has an error of 0.027. Table 2 shows the result of combining the operating system denoted “dos” with “windows” and that denoted “apple” with “mac”. Again, the classifier which calls all operating systems “windows” results in an error of 0.056 while this classifier has an error of 0.008. The file pass.R contains the R code to reproduce these results, and the details on which features were used. This code is provided as-is, with no warranties, guarantees, or claims.

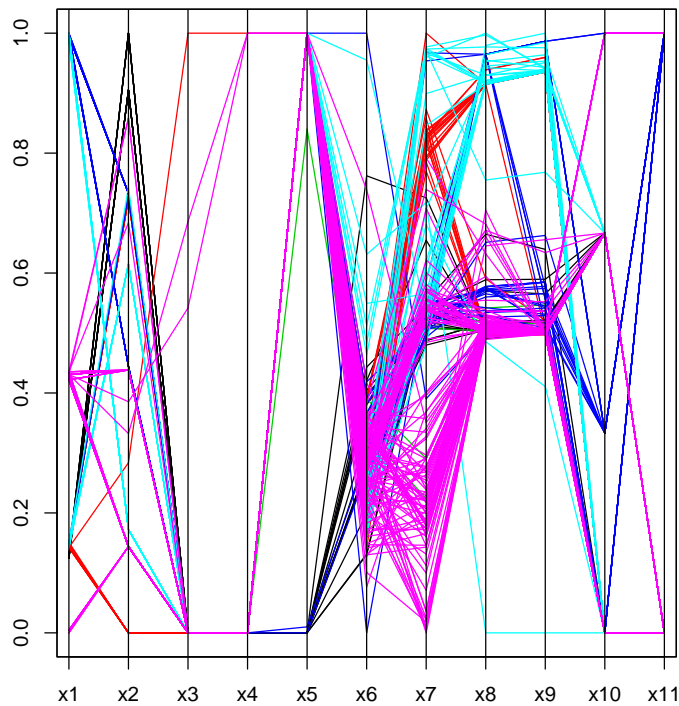


Figure 7: Parallel coordinates plot for some of the passive operating system data. The colors indicate the operating system.

These results indicate that statistical methods can be of value in passive fingerprinting. They are by no means the best possible. It should be noted that the accreditation database used to determine operating system may not have been completely accurate. The operating systems labeled “dos” and “apple” are particularly suspicious.

Table 1: Confusion matrix for the user profiling data. Red numbers indicate errors while blue indicate correct classification.

	dos	irix	linux	apple	mac	solaris	windows
dos	0	0	0	0	2	0	32
irix	0	16	0	0	0	0	1
linux	0	0	25	0	0	0	0
apple	0	0	0	0	3	0	3
mac	0	0	0	0	31	0	0
solaris	0	0	0	0	0	27	0
windows	1	0	6	0	3	0	1753

Table 2: Confusion matrix for the user profiling data, with the dos and apple classes collapsed to windows and mac respectively. Red numbers indicate errors while blue indicate correct classification.

	windows	irix	linux	mac	solaris
windows	1786	0	6	5	0
irix	1	16	0	0	0
linux	0	0	25	0	0
mac	3	0	0	34	0
solaris	0	0	0	0	27

## 4 User Profiling

Finally, we briefly look at one method for profiling user activity using network traffic. We will investigate the web servers visited by each user, and attempt to determine whether users tend to use the same servers, or if they have distinct patterns of surfing. This is a very preliminary study, merely to indicate some ideas of directions in which one might investigate.

Given the web servers visited by each user within a time period, the intersection graph is constructed as follows. The vertices correspond to the users, and there is an edge between two users if they both visited the same web server during this period. In this case, there is a 1-1 mapping from IP address to user, and so we can determine the user unambiguously from the IP address. This would not be the case for shared computers, as is typical in a university setting.

Figure 8 shows the graph for a period of 3 months. One could ask if the large clique size is unusual: do users tend to select the same web servers as other users, more than if they were selecting them at random? It is easy to see that if we believe the users to be selecting from all the possible web servers (several hundred thousand, if not millions of machines) that a clique of this size is unusual. However, consider the hypothesis that the set of servers observed is the full set from which these were drawn. This might be a reasonable assumption given that the users are all members of a single organization working on similar projects. There are several models we can use to assess the significance of the clique. We chose to consider the following: each user draws randomly (uniformly, iid) from the pool of servers, with the number chosen fixed to be the number we observed. Through Monte Carlo replication, we observe that we obtain cliques of size 26 or greater 62/100 times, with a minimum clique size of 24 and a maximum of 27 and so under this model this large a clique size is not at all unusual. Thus, we feel that conditional on the users selecting from this pool, we cannot reject the hypothesis that they are choosing randomly.

Figure 9 shows the graph for secure web servers. Using the same methodology, we observe 0/100 cliques of size at least 11, with a maximum clique size of 9 and a minimum of 6. This implies that (again with the condition that the users are drawing from the given pool) users tend to choose similar secure web servers. This might be reasonable, given that secure servers are more likely to have a specific purpose that is shared by a large group of people (for example, a server that provides telephone numbers for the company headquarters).

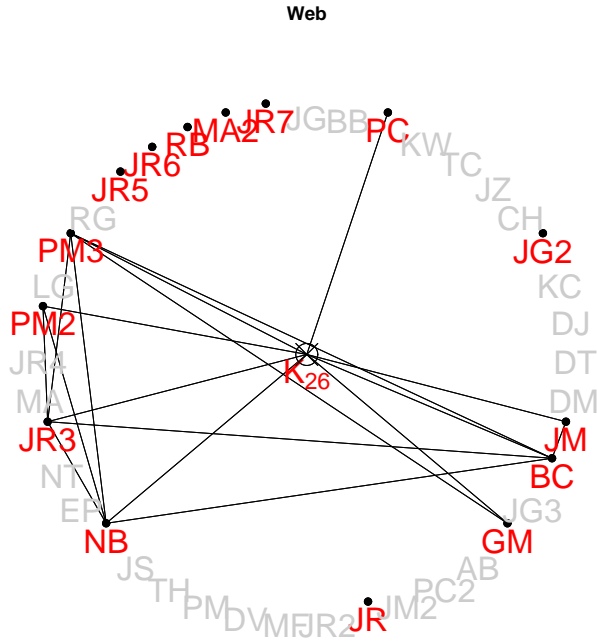


Figure 8: Intersection graph for web users. The clique of size 26 is shown in the center. Gray colored users in the circle correspond to users in the clique. The position of user node is consistent throughout the plots.

Finally, we look at the clique numbers for web server activity obtained on a weekly basis. For each week the intersection graph is computed on the data for that week, and the clique number is computed. This is plotted in Figure 10.

As can be seen, a change occurs between weeks 5 and 6. This is the period in which the fiscal year changes, indicating that the user’s behavior changes when the projects they are involved in change. Detecting these kinds of changes can be important for profiling the user behavior. While these examples are far too crude to be of use for detecting masqueraders, they illustrate interesting and useful methodologies. Investigating the statistics of these and related models can be interesting and rewarding.

## 5 Discussion

We have looked at three areas in which statistics can play a role in network monitoring and computer security. We deliberately chose areas that are novel and that the reader may not have come across in readings and conferences. Some areas that we have not touched on, and in which considerable work has been done, is anomaly detection, network tomography, worm and virus propagation, fraud detection and profiling users by keystroke timing or application usage. The interested reader can find information on these in Vardi [1996], Marchette [2001], Schonlau et al. [2001] and Bolton and Hand [2002].

Network data is quite complex. Many issues come to play in its analysis: proper representation of the data; visualization and exploratory data analysis; high data rates and large volume; non-stationarity; deterministic as well as random structure; multiple populations within the data. We have seen some of these in this paper. We feel that there are many more interesting problems to be addressed in network data, and in computer security data in general.



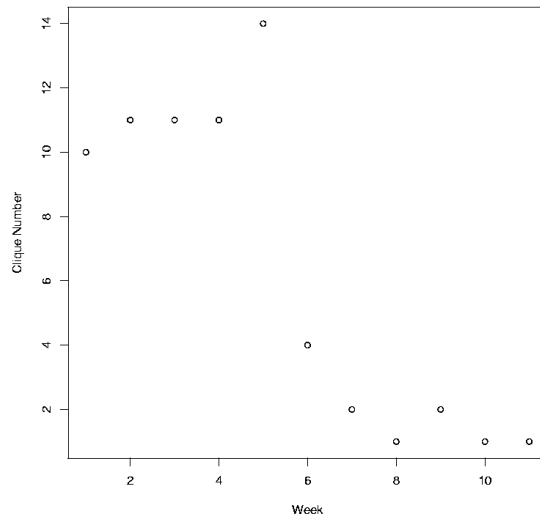


Figure 10: Clique numbers for intersection graphs built on data from each week. The week number is on the horizontal axis, the clique number on the vertical. These data start on September 1.