JGSnap: A Graphical Analyzer for Honeysnap

Thesis submitted in partial fulfillment of the requirements for the award of degree

Master of Engineering

in

Software Engineering

By:
Abhishek Vershney
(8053102)

Under the supervision of:
Dr. Maninder Singh
Assistant Professor
CSED, TU, Patiala

MAY 2007

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

Certificate

I hereby certify that the work which is being presented in the thesis entitled, ***"JGSnap: A Graphical Analyzer for Honeysnap"***, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my research work carried out under the supervision of *Dr. Maninder Singh*.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Abhishek Vershney)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Maninder Singh)
Assistant Professor
Computer Science & Engineering Department
Thapar University
Patiala – 147004

Countersigned By:

(Dr. (Mrs.) Seema Bawa)
Professor & Head
Computer Science & Engineering Department
Thapar University
Patiala – 147004

(Dr. R. K. Sharma)
Dean
Academic Affairs
Thapar University
Patiala – 147004

Acknowledgement

First of all, humble bow before the almighty God for his blessings and providing the power to finish the work.

I wish to express my deep gratitude to my supervisor Dr. Maninder Singh, Assistant Professor, Computer Science & Engineering Department, TU, Patiala for providing his uncanny guidance and support throughout the thesis.

I am also thankful to Dr. (Mrs) Seema Bawa, Professor and Head, Computer Science & Engineering Department, TU, Patiala for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank all the staff members and my co-students who were always there at the need of the hour and provided with all the help and facilities, which I required for the completion of the thesis.

Abhishek Vershney

Abstract

Internet it is rapidly changing its presence from a medium for elites to one in common use in our everyday lives. A decade ago, the first age of the Internet was a bright light shining above everyday concerns. It was a technological marvel bringing a new enlightenment to transform the world, just as the printing press fostered the original enlightenment a half millennium ago in Renaissance times. With the development of the Internet, and with the increasing pervasiveness of communication between networked computers, we are in the middle of the most transforming technological event since the capture of fire.

Even after having firewall, latest Intrusion Detection System, and antivirus system, our organizations' networks are not safe. If one patched antivirus for one worm, next day new worm comes up. By the time anyone patch network for this new worm, network may be attacked. So now the time it is very difficult to save our network from new attacks.

Proactive network security is the act of managing the different components of network security, e.g. firewall, VPNs, antivirus software etc, so that anyone get the most performance from them while at the same time augmenting system with a vulnerability management system. In this report main emphasis is given to services of Proactive Network Security, how to achieve proactive network security, tools and techniques that can be used to implement it

Honeypot plays an important role in implementing Proactive Security Approach. A honeypot is a closely monitored computing resource that intended to be probed, attacked, or compromised. The value of a honeypot is determined by the information that can be obtained from it. Monitoring the data that enters and leaves a honeypot lets us gather information that is not available to Network Intrusion Detection System.

In the market there are a lot of tools available for analyzing the data logged by Honeypot. These tools include Snort, Ethereal and Honeysnap. Honeysnap is a modular, python application that can parse raw or gzipped packet captured files and

performs a number of diagnostics on the data. Honeysnap comes in text flavor, means it performs all operation in text-based environment. Currently no Graphical User Interface is available for Honeysnap.

Apart from investigating proactive security realm, our objective is to design and develop a tool **JGSnap** that will provide a Graphical User Interface to Honeysnap, which will provide an easy to use interface to Honeysnap.

JGSnap is a Java-based data analyzer with a Web Graphical User Interface that provides the graphical analysis of data provided by Honeysnap. It can generate different reports for the data. One of its main characteristics is as it is being developed in Java; it can potentially run on any platform, it may be Windows, Linux or Solaris.

Table of Contents

Table of Figures

Chapter 1

Introduction

1.1 Relevance and Importance of Networks in Today's Life

Internet it is rapidly changing its presence from a medium for elites to one in common use in our everyday lives. A decade ago, the first age of the Internet was a bright light shining above everyday concerns. It was a technological marvel bringing a new enlightenment to transform the world, just as the printing press fostered the original enlightenment a half millennium ago in Renaissance times. With the development of the Internet, and with the increasing pervasiveness of communication between networked computers, we are in the middle of the most transforming technological event since the capture of fire.

We are moving from a world of, Internet wizards to a world of ordinary people, routinely using the Internet as an embedded part of their lives. It has become clear that the Internet is a very important thing. In fact, more people, in more countries, are using it more – in more different ways. The network is evolving into the backbone and, in many instances, the central nervous system of everyday life. No longer just a clunky, wire-based delivery system for various passive computer applications, the network has evolved into an active, often proactive, information and communication system that touches and interacts with almost every person and department everywhere--and beyond. [pa02]

According to the statistics given in the website http://www.internetworldstats.com/ 1,114,274,426[1] people are using the internet as according to the latest figures gathered in 2007.

The tables give us and idea how fast the Internet is growing. The number of people using the Internet is doubling every year. More sites pop up every day. Global communication is getting more important. More and more companies connect their

[1] Source http://www.internetworldstats.com/stats.htm

computer networks to the global Internet or connect multiple intranets over the Internet with the help of virtual private networks. E-Commerce is getting an important source of revenue for many companies. At the same time, computer crimes are increasing. Here are some of the trends discovered by the survey[2].

WORLD INTERNET USAGE AND POPULATION STATISTICS

World Regions	Population (2007 Est.)	Population % of World	Internet Usage, Latest Data	% Population (Penetration)	Usage % of World	Usage Growth 2000-2007
Africa	933,448,292	14.2 %	33,334,800	3.6 %	3.0 %	638.4 %
Asia	3,712,527,624	56.5 %	398,709,065	10.7 %	35.8 %	248.8 %
Europe	809,624,686	12.3 %	314,792,225	38.9 %	28.3%	199.5 %
Middle East	193,452,727	2.9 %	19,424,700	10.0 %	1.7 %	491.4 %
North America	334,538,018	5.1 %	233,188,086	69.7 %	20.9%	115.7 %
Latin America/Caribbean	556,606,627	8.5 %	96,386,009	17.3 %	8.7 %	433.4 %
Oceania / Australia	34,468,443	0.5 %	18,439,541	53.5 %	1.7 %	142.0 %
WORLD TOTAL	6,574,666,417	100.0 %	1,114,274,426	16.9 %	100.0 %	208.7 %

NOTES: (1) Internet Usage and World Population Statistics were updated on Mar. 10, 2007. (2) CLICK on each world region for detailed regional information. (3) Demographic (Population) numbers are based on data contained in the world-gazetteer website. (4) Internet usage information comes from data published by Nielsen//NetRatings, by the International Telecommunications Union, by local NICs, and other other reliable sources. (5) For definitions, disclaimer, and navigation help, see the Site Surfing Guide. (6) Information from this site may be cited, giving due credit and establishing an active link back to www.internetworldstats.com. Copyright © 2007, Miniwatts Marketing Group. All rights reserved worldwide.

Figure 1.1: World Internet Usage [in03]

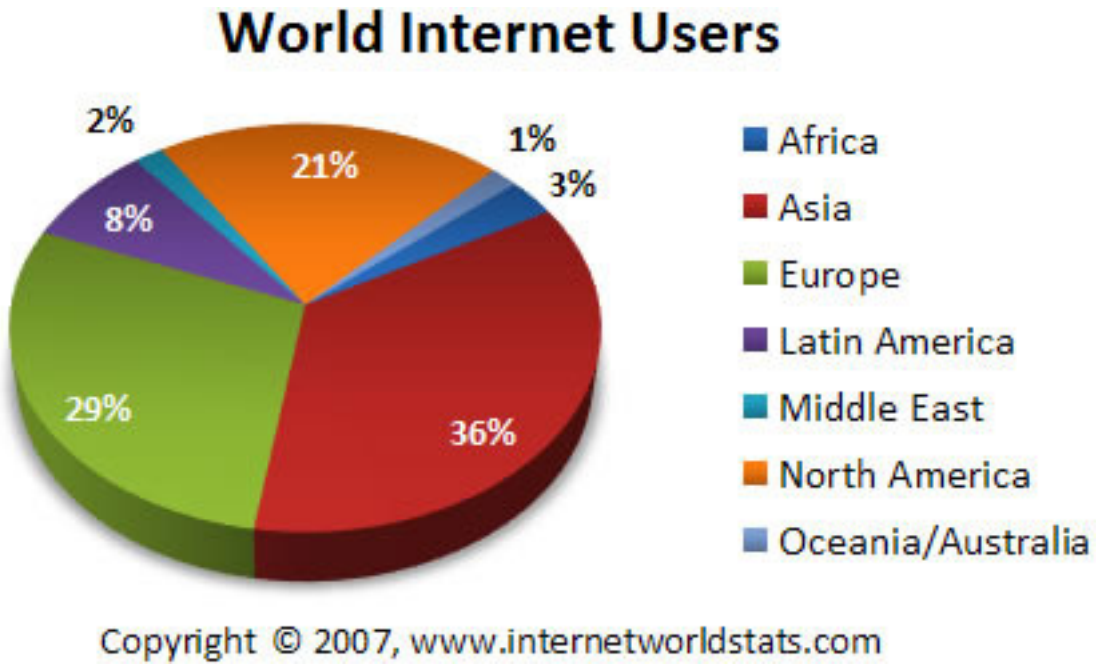


Figure 1.2: World Internet Users % [in03]

[2] Source http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2006.pdf, Courtesy of Robert Richardson, Director Computer Security Institute.

- Virus attacks continue to be the source of the greatest financial losses. Unauthorized access continues to be the second-greatest source of financial loss. Financial losses related to laptops (or mobile hardware) and theft of proprietary information (i.e., intellectual property) are third and fourth. These four categories account for more than 74 percent of financial losses.

- Unauthorized use of computer systems slightly decreased this year, according to respondents.

- The total dollar amount of financial losses resulting from security breaches had a substantial decrease this year, according to respondents. Although a large part of this drop was due to a decrease in the number of respondents able and willing to provide estimates of losses, the average amount of financial losses per respondent also decreased substantially this year.

- Despite talk of increasing outsourcing, the survey results related to outsourcing are similar to those reported in the last two years and indicate very little outsourcing of information security activities. In fact, 61 percent of the respondents indicated that their organizations do not outsource any computer security functions. Among those organizations that do outsource some computer security activities, the percentage of security activities outsourced is rather low.

INTERNET USERS AND POPULATION STATISTICS FOR ASIA

ASIA REGION	Population (2007 Est.)	% Pop. of World	Internet Users, Latest Data	Penetration (% Population)	% Usage of World	Use Growth (2000-2007)
Asia Only	3,712,527,624	56.5 %	398,709,065	10.7 %	35.8 %	248.8 %
Rest of the World	2,862,138,793	43.5 %	715,565,361	25.0 %	64.2 %	190.1 %
WORLD TOTAL	6,574,666,417	100.0 %	1,114,274,426	16.9 %	100.0 %	208.7 %

NOTES: (1) Asiatic Internet Usage and Population Statistics were updated on Mar. 10, 2007. (2) Population numbers are based on data contained in world gazetteer. (3) The most recent usage comes mainly from data published by Nielsen//NetRatings , ITU , and other local sources. (4) Data on this site may be cited, giving due credit and establishing an active link back to Internet World Stats . (5) For definitions and help, see the site surfing guide. Copyright © 2007, Miniwatts Marketing Group. All rights reserved.

Figure 1.3: Asia Internet Users [in03]

- Once again, the vast majority of the organizations view security awareness training as important. In fact, there is a substantial increase in the respondents' perception of the importance of security awareness training. On average,

respondents from most sectors do not believe their organization invests enough in this area.

At the same time countermeasures are getting more sophisticated and widely used - firewalls pop up in nearly every company and people at home install so called “personal firewalls”. Network intrusion detection systems start their triumphal procession. All these countermeasures are based on a defending, detection and reaction mechanism. But do we know enough of our enemy or are these systems always a step behind the blackhats.

As in the military, it is important to know, who your enemy is, what kind of strategy they use, what tools they have and what they are aiming for. Gathering this kind of information is not easy but important. By knowing attack strategies, countermeasures can be taken and vulnerabilities can be fixed. To gather as much information as possible is one main goal of a honeypot.

Generally, such information gathering should be done silently, without alarming an attacker. All the gathered information leads to an advantage on the defending side and can therefore be used on productive systems to prevent attacks. A production honeypot is one used within an organization's environment to help mitigate risk; it has value to the security of production resources. A research honeypot also called a Honeynet is a collection of more than one honeypots working in unison. Specifically, it is a high-interaction honeypot designed to capture extensive information on threats. Highinteraction means a Honeynet provides real systems, applications, and services for attackers to interact with (as opposed to low-interaction honeypots such as Honeyd, which provide emulated services and operating systems).

A honeypot is primarily an instrument for information gathering and learning. Its primary purpose is not to be an ambush for the blackhat community to catch them in action and to press charges against them. The focus lies on a silent collection of as much information as possible about their attack patterns, used programs, purpose of attack and the blackhat community itself. All this information is used to learn more about the blackhat proceedings and motives, as well as their technical knowledge and abilities. But this is just a primary purpose of a honeypot. There are a lot of other

possibilities to use a honeypot for, to divert hackers from productive systems or to catch a hacker while conducting an attack.

In the right hands, a honeypot can be an effective tool for information gathering. This happens as honeypots mostly allow all inbound traffic to come to them, sitting behind a reverse firewall. And in order to mitigate risk the outbound traffic is restricted.

1.2 Need of Network Security

While using the Internet, along with the convenience and speed of access to information new risks arise. Among them are the risks that valuable information will be lost, stolen, corrupted, or misused and that the computer systems will be corrupted. If information is recorded electronically and is available on networked computers, it is more vulnerable than if the same information is printed on paper and locked in a file cabinet. Intruders do not need to enter an office or home, and may not even be in the same country. They can steal or tamper with information without touching a piece of paper or a photocopier. They can create new electronic files, run their own programs, and even hide all evidence of their unauthorized activity. [wi01]

It is remarkably easy to gain unauthorized access to information in an insecure networked environment, and it is hard to catch the intruders. Even if users have nothing stored on their computer that they consider important, that computer can be a "weak link", allowing unauthorized access to the organization's systems and information. Seemingly innocuous information can expose a computer system to compromise. Information that intruders find useful includes which hardware and software are being used, system configuration, type of network connections, phone numbers, and access and authentication procedures. Security-related information can enable unauthorized individuals to get access to important files and programs, thus compromising the security of the whole system. Today's commercial off-the-shelf [software] technology is riddled with holes. The sheer number of vulnerabilities is overwhelming organizations. These include vulnerabilities that allowed viruses and worms (hereafter referred to as malware) and other manual and automated attacks to inflict damages costing hundreds of millions of dollars per occurrence. Specifically:

LoveLetter, a worm that severely clogged mail servers and networks in 2000; Code Red, an aggressive worm that attacked unpatched Microsoft web servers and defaced their main pages; and most recently, Nimda, a worm that spread by several different methods including email and web protocols, and searched for as many as 16 separate vulnerabilities to attack. The recent Distributed Denial of Service (DDOS) attacks are less serious but still expensive. Virus attacks, exploits directed at unpatched popular firewalls (e.g., Check Point, Cisco Pix), buffer overflows, directory traversal, other more obscure attacks against web servers, and the scope of the problem starts to become quite clear

Those affected include banks and financial companies, insurance companies, brokerage houses, consultants, government contractors, government agencies, hospitals, medical laboratories, network service providers, utility companies, the textile industry, universities, wholesale and retail trades. The consequences of a break-in cover a broad range of possibilities: a minor loss of time in recovering from the problem, a decrease in productivity, a significant loss of money or staff-hours, a devastating loss of credibility or market opportunity, a business no longer able to compete, legal liability, and the loss of life.

1.3 Various Threats to Network Security

1.3.1 Viruses

A computer virus is a special kind of computer program which: Spreads across disks and networks by making copies of itself, usually surreptitiously and / or can produce undesired side-effects in computers in which it is active.
There are various types of viruses:

- Boot viruses place (some of) their code in the disk sector whose code the machine will automatically execute when booting. Thus, when an infected machine boots, the virus loads and runs. After boot viruses are finished loading, they usually load the original boot code, which they have previously moved to another location, or take other measures to ensure the machine appears to boot normally.

- File viruses attach to ‘program files’ (files containing executable or interpretable code) in such a way that when infected program is run, the virus code executes. Usually the virus code is added in such a way that it executes first, although this is not strictly necessary. After the virus code has finished loading and executing, it will normally load and execute the original program it has infected, or call the function it intercepted, so as to not arouse the user’s suspicion.

- Macro viruses are really just a type of file virus, but a particularly ‘successful’ type. They copy their macros to templates and/or other application document files. Although ‘auto macros’ were almost exclusively used by early macro viruses (often to ensure the virus’ code is the first to execute when infected templates or documents were opened), several other mechanisms are also available – in fact, some of these, such as taking over standard internal functions of the host application (say the ‘File Save’ command) and installing default event handlers are probably more commonly used these days.

- Script viruses also became quite successful around the beginning of this century. This was mainly due to the increase in machines running Windows Scripting Host, which was first installed by default in Windows 98 and 2000 and with Internet Explorer 5.0 and later versions. Representing new types of ‘program file’, but with icons more like that of ‘safe’ text files, standalone Visual Basic Script (VBS) and JavaScript (JS) programs became a popular target of the writers of mass mailing viruses.

- Companion viruses take advantage of features of the operating system to be executed, rather than directly infecting programs or boot sectors. Under DOS and Windows, when user executes the command ‘ABC’, the rule is that ABC.COM executes before ABC.EXE (in the rare cases where both files exist). Thus, a companion virus could place its code in a COM file with its first name matching that of an existing EXE file. When the user next executed the ‘ABC’ command, the virus’ ABC.COM program would be run (usually

the virus would launch ABC.EXE once its code was finished so as not to arouse suspicion). This is known as the 'execution preference companion' method, but several other forms of companion infection are also possible.

- Worms are described by some antivirus researchers as similar to viruses in that they make copies of themselves, but different in that they need not attach to particular files or sectors at all. Once such a worm is executed, it seeks other systems – rather than parts of systems – to infect, then copies its code to them in such a way as to have the code execute directly from memory. More recently the term 'worm' has been taken to mean 'a virus that replicates across a network link', with the most common usage applied to viruses that send many copies of themselves out attached to the infected user's e-mail.

Some viruses display obvious symptoms, and some cause damage to files in a system they have infected. While one or both of these features of a virus often capture the attention of the popular media, note from the preceding discussion that neither are essential in the definition of a virus. A non-damaging virus is still a virus, not a prank and, other things being equal, viruses without obvious symptoms are more likely to spread further and persist longer than those that rapidly draw attention to themselves.

1.3.2 Trojan horse programs

Just as the mythological Trojan horse appeared to be a gift, but turned out to contain Greek soldiers who overtook the city of Troy, today's Trojan horses are computer programs that appear to be useful software, but instead they compromise the security and cause a lot of damage. A recent Trojan horse came in the form of an e-mail that included attachments claiming to be Microsoft security updates, but turned out to be viruses that attempted to disable antivirus and firewall software. Trojan horses spread when people are lured into opening a program because they think it comes from a legitimate source. Trojan horses can also be included in software that downloaded for free. Never download software from a source that are not trusted. There are six ways Trojans generally capture passwords

1. INTERRUPT 9h. This is a hardware interrupt which occurs every time one hits a key. The Trojan waits for an INT 9h, and then interrogates the keyboard to discover which key was pressed, saves the key, and then returns normal control.
2. INTERRUPT 16h. This is an operating system call which deals with the keyboard. Every time an application program needs a key, it calls this software interrupt. A Trojan piggybacks onto the INT 16h and saves the keys every time the program asks for them.
3. INTERRUPT 21h. Similar to INT 16h, much more widely known, but less elegant. Once again, a Trojan will piggyback INT 21h.
4. INTERRUPT 15h. Similar to INT 16h, but called from INT 9h. A Trojan will again piggyback to this BIOS function in order to capture passwords.
5. Hardware polling. This method simply polls the keyboard many times a second, and records any new keys press as it notices user pressing them. If user types too fast, it will miss some.
6. RAM-buffer snooping. A Trojan can copy the DOS keyboard type-ahead buffer from the system area (also called segment 40:) or from the executable-programs memory regularly to disk.

Once the password is captured, the Trojan will usually either save it in RAM, or on a local or network disk (in a hidden file in a hidden directory), or even in CMOS. A purpose-written Trojan can be programmed to download passwords via modem or onto the internet, or broadcast them to some other place for remote retrieval.

Some Trojans activate only when a password is being entered. They do this by monitoring system calls, such as the "execute program" call when LOGIN.EXE is being run. Others "read" the screen and record when they see some phrase, such as "password" appear. Many others simply record everything.

A Trojan is one of the most dangerous tools available to network intruders. The most obvious reason for the danger is that a Trojan provides the intruder with a guaranteed method of access to the network, under the guise of a legitimate user. Because the Trojan provides the intruder with a legitimate, valid username and password to access the network with, traditional methods of detecting and preventing this intrusion are

useless. An intruder with a Trojan will be logging straight into the network without setting off any intrusion alarms from guessed password attempts or suspicious log-in activity.

The Trojan provides the means for the intruder to identify how to log in, and when and where it is safest to do so without being detected. Since the network has no way at all to distinguish between the intruder and the legitimate user, this access will be undetectable as well as undetected. Many network administrators believe that because they employ anti-virus software, they are not at risk. This couldn't be further than the truth. In fact, this misconception actually amplifies the power of a Trojan to do damage. Most Trojans are not viral, and hence can never be detected by anti-virus software. Further — since a Trojan can even be a legitimate program, there is no way that antivirus software can address this threat in the future, should they even attempt to, since the "signatures" would be copyright.

So the false piece of mind generated by a "clean" scan of a disk, system, or network is only serving to foster ignorance of the threat. Unlike viruses, no-one has a library of Trojans, and so most are unknown in nature. Further, since a Trojan is far smaller and simpler than even a basic virus, it much easier to write. Worse still, a Trojan is, by design and nature, a silent and secret threat. It does not introduce itself to its victim, and so goes unnoticed in most cases. Finally, as mentioned above, even the operation of a Trojan is generally unnoticed, reducing the chances of the Trojan ever being discovered to minuscule proportions.

1.3.3 Attacks

Including reconnaissance attacks (information-gathering activities to collect data that is later used to compromise networks); access attacks (which exploit network vulnerabilities in order to gain entry to e-mail, databases, or the corporate network); and denial-of-service attacks (which prevent access to part or all of a computer system).

1.3.3.1 Reconnaissance Attack

Reconnaissance is the unauthorized discovery and mapping of systems, services, or vulnerabilities. It is also called information gathering. In most cases, it precedes an actual access or DoS attack. The malicious intruder typically ping-sweeps the target network first to determine what IP addresses are alive. After this is accomplished, the intruder determines what services or ports are active on the live IP addresses. From this information, the intruder queries the ports to determine the application type and version as well as the type and version of the operating system running on the target host.

Reconnaissance is somewhat analogous to a thief scoping out a neighborhood for vulnerable homes he can break into, such as an unoccupied residence, an easy-to-open door or window, and so on. In many cases, an intruder goes as far as "rattling the door handle" not to go in immediately if it is open, but to discover vulnerable services he can exploit later when there is less likelihood that anyone is looking.

1.3.3.2 Access Attack

Access is an all-encompassing term that refers to unauthorized data manipulation, system access, or privilege escalation. Unauthorized data retrieval is simply reading, writing, copying, or moving files that are not intended to be accessible to the intruder. Sometimes this is as easy as finding shared folders in Windows 9*x* or NT, or NFS exported directories in UNIX systems with read or read-write access to everyone. The intruder has no problem getting to the files. More often than not, the easily accessible information is highly confidential and completely unprotected from prying eyes, especially if the attacker is already an internal user.

System access is an intruder's ability to gain access to a machine that he is not allowed access to (such as when the intruder does not have an account or password). Entering or accessing systems that user doesn't have access to usually involve running a hack, script, or tool that exploits a known vulnerability of the system or application being attacked.

Another form of access attacks involves privilege escalation. This is done by legitimate users who have a lower level of access privileges or intruders who have

gained lower-privileged access. The intent is to get information or execute procedures that are unauthorized at the user's current level of access. In many cases this involves gaining root access in a UNIX system to install a sniffer to record network traffic, such as usernames and passwords that can be used to access another target.

1.3.3.3 Denial of Service Attack

DoS (Denial-of-Service) attacks are probably the nastiest, and most difficult to address. These are the nastiest, because they're very easy to launch, difficult (sometimes impossible) to track, and it isn't easy to refuse the requests of the attacker, without also refusing legitimate requests for service.

The premise of a DoS attack is simple: send more requests to the machine than it can handle. There are toolkits available in the underground community that make this a simple matter of running a program and telling it which host to blast with requests. The attacker's program simply makes a connection on some service port, perhaps forging the packet's header information that says where the packet came from, and then dropping the connection. If the host is able to answer 20 requests per second, and the attacker is sending 50 per second, obviously the host will be unable to service all of the attacker's requests, much less any legitimate requests (hits on the web site running there, for example).

Obviously forged packets would include those that claim to come from your own hosts, addresses reserved for private networks as defined in RFC 1918 and the *loopback* network (127.0.0.0).

1.3.4 Data interception

Data transmitted via any type of network can be subject to interception by unauthorized parties. The intercepting perpetrators might eavesdrop on communications or even alter the data packets being transmitted. Perpetrators can use various methods to intercept data. IP spoofing, for example, entails posing as an

authorized party in the data transmission by using the Internet Protocol (IP) address of one of the data recipients.

One of the most widespread method of Data interception is “Man in the Middle Attack (MITM)”. In cryptography, a *man in the middle attack (MITM)* is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised. The attacker must be able to observe and intercept messages going between the two victims.

Suppose ‘A’ wishes to communicate with ‘B’, and that ‘C’ wishes to eavesdrop on the conversation, or possibly deliver a false message to ‘B’. To get started, ‘A’ must ask ‘B’ for his public key. If ‘B’ sends his public key to ‘A’, but ‘C’ is able to intercept it, a man in the middle attack can begin. ‘C’ can simply send ‘A’ a public key for which he has the private, matching, key. ‘A’, believing this public key to be ‘B’'s, then encrypts her message with ‘C’'s key and sends the encyphered message back to ‘B’. ‘C’ again intercepts, decyphers the message, keeps a copy, and reencyphers it (after alteration if desired) using the public key ‘B’ originally sent to ‘A’. When ‘B’ receives the newly encyphered message, he will believe it came from ‘A’. A similar attack is possible, in principle, against any message sent using public key technology, including data packets carried on computer networks.

1.3.5 Social engineering

Social Engineering is an attack method used by many attackers that takes advantage of trust and complacency at work. Humans by nature are very trusting and rarely question actions that are considered normal. Another forum that Social Engineering can expose is the Computer Conference. Computer conferences are great for obtaining information. Most conferences stress openness, this within itself is not a bad idea but the problem occurs when people give too many details. Some of the information that attendee's and instructors give out could be used against them and their network(s). Information about network configuration, types of firewalls and Intrusion Detection systems were just a few items commonly shared.

Dumpster diving, also known as trashing, is another popular method of social engineering. A huge amount of information can be collected through company dumpsters. The following items as potential security leaks in our trash: "company phone books, organizational charts, memos, company policy manuals, calendars of meetings, events and vacations, system manuals, printouts of sensitive data or login names and passwords, printouts of source code, disks and tapes, company letterhead and memo forms, and outdated hardware."

Other major types of Social Engineering attacks used are *Social Engineering by Phone*, *Persuasion* etc.

1.3.6 Vandals

Web sites have come alive through the development of such software applications as ActiveX and Java Applets. These applications enable animation and other special effects to run, making web sites more attractive and interactive. However, the ease with which these applications can be downloaded and run has provided a new vehicle for inflicting damage. Vandals can take on the form of a software application or applet that causes destruction of various degrees. A vandal can destroy a single file or a major portion of a computer system.

1.4 Categories of Network Security

Network Security and Services can be categorized as follow:

- Reactive Network Security and Services
- Proactive Network Security and Services

In Reactive Network Security, security services are triggered by an event or request, such as a report of a compromised host, wide-spreading malicious code, software vulnerability, or something that was identified by an intrusion detection or logging system.

Reactive Network Security services are designed to respond to requests for assistance, and any threats or attacks against systems. Some services may be initiated by third-party notification or by viewing monitoring or IDS logs and alerts.

Example of Reactive Network Security services are Alerts and Warnings, Incident Handling, Vulnerability Handling etc.

1.5 Proactive Network Security and Services

Proactive Network Security is the act of managing the different components of network security, e.g. firewall, VPNs, antivirus software etc, so that one get the most performance from them while at the same time augmenting one's system with a vulnerability management system. A more effective firewall is going to block the right traffic. A more effective antivirus program is going to have less work to do, because viruses will have fewer opportunities to attack our systems. But preventing the attack with a vulnerability management system to eliminate Common Vulnerabilities and Exposures is the most important component.

A proactive system constantly *tests* the organization's network for vulnerabilities and exposures. It then *assesses and prioritizes* those vulnerabilities and exposures and *manages* the process by which those vulnerabilities and exposures are addressed. All IP devices attached to the network are periodically or continuously scanned and profiled for changes, violations to policy, and vulnerabilities and exposures. Analytics are applied so that the administrators and business owners are presented with actionable intelligence relative to the risk to their business. The defect is then corrected, before security can be breached.
Why Proactive Network Security? According to the many surveys 95 percent of all security breaches result from known vulnerabilities and misconfigurations. In reality, it just plain makes more sense to lock the doors and keep intruders out than to solve the problems after intruders have already broken in. Anyone wouldn't leave one's house unlocked, so why leave our network unlocked? [ga01]

Proactive Network Security services are designed to improve the infrastructure and security processes of the constituency before any incident or event occurs or is

detected. The main goals are to avoid incidents and to reduce their impact and scope when they do occur.

Following are services those are provided in Proactive Network Security.

1.5.1 Security Audits or Assessments

This service provides a detailed review and analysis of an organization's security infrastructure, based on the requirements defined by the organization or by other industry standards that apply. It can also involve a review of the organizational security practices. [ga01] There are many different types of audits or assessments that can be provided, including

- ✓ Infrastructure review
- ✓ Best practice review
- ✓ Scanning
- ✓ Penetration Testing

Obtaining upper management approval is required before conducting such audits or assessments since ome of these approaches may be prohibited by organizational policy.

1.5.2 Configuration and Maintenance of Security Tools, Applications, Infrastructures, and Services

This service identifies or provides appropriate guidance on how to securely configure and maintain tools, applications, and the general computing infrastructure used by the security team. Besides providing guidance, the security team may perform configuration updates and maintenance of security tools and services, such as IDS, network scanning or monitoring systems, filters, wrappers, firewalls, virtual private networks (VPN), or authentication mechanisms. The security team may even provide these services as part of their main function. The security team may also configure and maintain servers, desktops, laptops, personal digital assistants (PDAs), and other wireless devices according to security guidelines. This service includes escalating to management any issues or problems with configurations or the use of tools and applications that the security team believes might leave a system vulnerable to attack.

1.5.3 Development of Security Tools

This service includes the development of any new, constituent-specific tools that are required or desired by the constituency or by the security team. This can include, for example, developing security patches for customized software used by the constituency or secured software distributions that can be used to rebuild compromised hosts. It can also include developing tools or scripts that extend the functionality of existing security tools, such as a new plug-in for a vulnerability or network scanner, scripts that facilitate the use of encryption technology, or automated patch distribution mechanisms.

1.5.4 Analyzing IDS Logs

Security team that perform this service review existing IDS logs, analyze and initiate a response for any events that meet their defined threshold, or forward any alerts according to a pre-defined service level agreement or escalation strategy. Intrusion detection and analysis of the associated security logs can be a daunting task—not only in determining where to locate the sensors in the environment, but collecting and then analyzing the large amounts of data captured. In many cases, specialized tools or expertise is required to synthesize and interpret the information to identify false alarms, attacks, or network events and to implement strategies to eliminate or minimize such events. Some organizations choose to outsource this activity to others who have more expertise in performing these services, such as managed security service providers.

1.5.5 Security-Related Information Dissemination

This service provides constituents with a comprehensive and easy-to-find collection of useful information that aids in improving security. Such information might include

- ✓ reporting guidelines and contact information for the security team
- ✓ archives of alerts, warnings, and other announcements
- ✓ documentation about current best practices
- ✓ general computer security guidance
- ✓ policies, procedures, and checklists
- ✓ patch development and distribution information

- ✓ vendor links
- ✓ current statistics and trends in incident reporting
- ✓ other information that can improve overall security practices

This information can be developed and published by the security team, administrator or by another part of the organization (IT, human resources, or media relations), and can include information from external resources such as other security teams, vendors, and security experts. [ce01]

Chapter 2

Literature Survey

Honeypots are systems used to lure hackers by exposing known vulnerabilities deliberately. Once a hacker finds a honey pot, it is more likely that the hacker will stick around for some time. During this time hacker activities can be logged to find out hacker's actions and techniques. Once anyone knows these techniques, this information can be used later on to harden security on the actual servers.

L. Spitzner[3] defines the term Honeypot as follows:

"…A honeypot is a resource whose value is being in attacked or compromised. This means that a Honeypot is to get probed, attacked and potentially exploited. Honeypot do not fix anything. They provide us with additional, valuable information. [se04]"

A honeypot is a closely monitored computing resource that intended to be probed, attacked, or compromised. The value of a honeypot is determined by the information that can be obtained from it. Monitoring the data that enters and leaves a honeypot lets us gather information that is not available to NIDS. For example, user can log the keystrokes of an interactive session even if encryption is used to protect the network traffic. To detect malicious behavior, NIDS require signatures of known attacks and often fail to detect compromises that were unknown at the time it was deployed. On the other hand, honeypots can detect vulnerabilities that are not yet understood. For example, user can detect compromise by observing network traffic leaving the honeypot even if the means of the exploit has never been seen before.

Because a honeypot has no production value, any attempt to contact it is suspicious. Consequently, forensic analysis of data collected from honeypots is less likely to lead to false positives than data collected by NIDS. [ja01]

[3] Lance Spitzner is a key member of a research group in the United States called Project Honeynet (http://project.honeynet.net).

Honeypots can run any operating system and any number of services. The configured services determine the vectors available to an adversary for compromising or probing the system. On the basic of services, Honeypot can be categories as,

- High-Interaction Honeypot
- Low-Interaction Honeypot

A High- interaction honeypot simulates all aspects of an operating system. A low-interaction honeypots simulates only some parts, for example the network stack. A high-interaction honeypot can be compromised completely, allowing an adversary to gain full access to the system and use it to launch further network attacks. In contrast, low-interaction honeypots simulate only services that cannot be exploited to get complete access to the honeypot. Low-interaction honeypots are more limited, but they are useful to gather information at a higher level, e.g., learn about network probes or worm activity. They can also be used to analyze spammers or for active countermeasures against worms.

Honeypot can also be categorized as physical and virtual honeypot. A physical honeypot is a real machine on the network with its own IP address. A virtual honeypot is simulated by another machine that responds to network traffic sent to the virtual honeypot.

When gathering information about network attacks or probes, the number of deployed honeypots influences the amount and accuracy of the collected data. A good example is measuring the activity of HTTP based worms. These worms can only be identified after they complete a TCP handshake and send their payload. However, most of their connection requests will go unanswered because they contact randomly chosen IP addresses. A honeypot can capture the worm payload by configuring it to function as a web server. The more honeypots is deployed the more likely one of them is contacted by a worm. Physical honeypots are often high-interaction, so allowing the system to be compromised completely, they are expensive to install and maintain. For large address spaces, it is impractical or impossible to deploy a physical honeypot for each IP address. In that case, virtual honeypots needs to be deployed.

Risk mitigation

A honeypot deployed in a productive environment may lure an attacker away from the real production systems. A Honeypot is unpatched and vulnerable system and so an easy target for an attacker. An attacker will try to enter into the network through an easy target not a fully patched system. This will make our real production system safe.

IDS-like functionality:

Since no legitimate traffic should take place to or from the honeypot, any traffic appearing is evil and can initiate further actions. This traffic can further analyze to determine what IP is most vulnerable for our network, which virus, Trojan is trying to enter in our network and so on.

Attack strategies:

Identify reasons and strategies why and how the system was attacked. This was the Honeypot, which doesn't have any useful information but the same attack can be launched against real production network. Through this traffic, all reason of attack, vulnerabilities needs to be identify so that they can be patched to save real production network.

Identification and classification:

The traffic can be analyzed to determine who is attacking, who is trying to probe our network. This may be in form of some IP addresses, worms, Trojans or something else. Through this data, origin of attacker can be determined.

Evidence

Once the attacker is identified all data captured may be used in a legal procedure. This traffic can be used as the proof against the attacker.

Increased knowledge:

By knowing how user is attacked user is able to enlarge your ability to respond in an appropriate way and to prevent future attacks.

Research:

Operating and monitoring a honeypot can reveal most up-to-date techniques/exploits and tools used as well as internal communications of the hackers or infection or spreading techniques of worms or viruses.

2.1 Honeypots Examples

In the market there are lot honeypot software or software based on honeypot concept are available. Some of them are given below [ho01]:

Honeyd

Honeyd is a small daemon that creates virtual hosts on a network. The hosts can be configured to run arbitrary services, and their personality can be adapted so that they appear to be running certain operating systems. Honeyd enables a single host to claim multiple addresses on a LAN for network simulation. Honeyd improves cyber security by providing mechanisms for threat detection and assessment. It also deters adversaries by hiding real systems in the middle of virtual systems. [ho02]

Honeyd for Windows

Windows port of the popular Honeyd software. Honeyd-win32 has all the capabilities of the UNIX version of honeyd with the exception of subsystems. Scripts, proxies, etc are all 100% supported.

Honeywall

The Honeywall CDROM combines all the tools and requirements of a Honeynet gateway on an easy to use, bootable CDROM. The intent is to make honeynets easier to deploy and customize. Simply boot off the CDROM, configure it based on the environment, and have a Honeywall gateway ready to go. The CDROM supports several configuration methods, including an interactive menu and .iso customization scripts. The CDROM is an appliance, based on a minimized and secured Linux OS. [ho03]

HoneyBOT

HoneyBOT is a Windows based medium interaction honeypot solution. HoneyBOT works by opening over 1000 udp and tcp listening sockets on a computer and these sockets are designed to mimic vulnerable services. When an attacker connects to these services they are fooled into thinking they are attacking a real server. The honeypot safely captures all communications with the attacker and logs these results for future analysis. An attacker attempts an exploit or uploads a rootkit or trojan to the server, the honeypot environment will safely store these files on the computer for analysis and submission to antivirus vendors. [ho04]

Back Officer Friendly

Back Officer Friendly was originally created to detect when anyone attempts a Back Orifice scan against a computer. It has since evolved to detect attempted connections to other services, such as Telnet, FTP, SMTP, POP3 and IMAP2. When BOF receives a connection to one of these services, it will fake replies to the hopeful hacker, wasting the attacker's time, and giving us time to stop them from other mischief. [ba01].

ProxyPot

An open proxy honeypot (proxypot) is a server that pretends to be an open proxy, taking requests from bad people to do bad things, and responding with a simulation instead of doing the evil deed. The goal is to fool the bad people into thinking they've done their bad thing and got away with it, while actually they didn't do it, and they got caught anyway. The proxypot found here is designed primarily to catch one kind of Internet bad guy: the mail spammer. [pr01]

KFSensor

KFSensor is a Windows based honeypot Intrusion Detection System (IDS). It acts as a honeypot to attract and detect hackers and worms by simulating vulnerable system services and trojans. By acting as a decoy server it can divert attacks from critical

systems and provide a higher level of information than can be achieved by using firewalls and NIDS alone. KFSensor is designed for use in a Windows based corporate environment and contains many innovative and unique features such as remote management, a Snort compatible signature engine and emulations of Windows networking protocols. With its GUI based management console, extensive documentation and low maintenance, KFSensor provides a cost effective way of improving an organization's network security. [kf02]

2.2 Data Analyzing Tools

2.2.1 Snort

Snort is a free, open source network intrusion detection and prevention system capable of performing packet logging and real-time traffic analysis, on IP networks. Snort was written by Martin Roesch but is now owned and developed by Sourcefire, of which Roesch is the founder and current CTO. Proprietary versions with integrated hardware and support services are sold by Sourcefire.

Snort is capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, OS fingerprinting attempts, amongst other features. The system can also be used for intrusion prevention purposes, by dropping attacks as they are taking place. [in02]

Components of Snort

Snort is logically divided into multiple components. These components work together to detect particular attacks and to generate output in a required format from the detection system. A Snort-based IDS consists of the following major components:

• Packet Decoder
• Preprocessors
• Detection Engine

• Logging and Alerting System
• Output Modules

Figure 2.1 shows how these components are arranged. Any data packet coming from the Internet enters the packet decoder. On its way towards the output modules, it is either dropped, logged or an alert is generated.

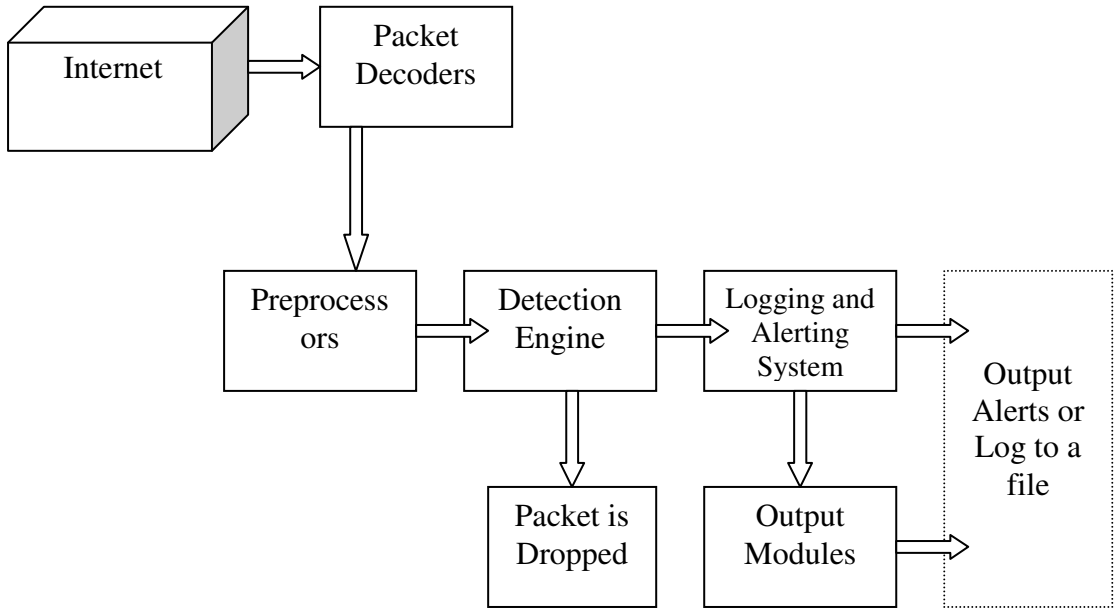


Figure 2.1: Snort Architecture [in02]

A brief introduction to these components is presented in the following section.

Packet Decoder

The packet decoder takes packets from different types of network interfaces and prepares the packets to be preprocessed or to be sent to the detection engine. The interfaces may be Ethernet, SLIP, PPP, and so on.

Preprocessors

Preprocessors are components or plug-ins that can be used with Snort to arrange or modify data packets before the detection engine does some operation to find out if the packet is being used by an intruder. Some preprocessors also perform detection by finding anomalies in packet headers and generating alerts. Preprocessors are very important for any IDS to prepare data packets to be analyzed against rules in the detection engine. Hackers use different techniques to fool IDS in different ways. For example, a rule have been created to find a signature "scripts/iisadmin" in HTTP packets. If this string is being matched exactly, a hacker who makes slight modifications to this string, can make fool easily. For example:

- "scripts/./iisadmin"
- "scripts/examples/../iisadmin"

To complicate the situation, hackers can also insert in the web Uniform Resource Identifier (URI) hexadecimal characters or Unicode characters which are perfectly legal as far as the web server is concerned. Note that the web servers usually understand all of these strings and are able to preprocess them to extract the intended string "scripts/ iisadmin". However if the IDS is looking for an exact match, it is not able to detect this attack. A preprocessor can rearrange the string so that it is detectable by the IDS.

Preprocessors are also used for packet defragmentation. When a large data chunk is transferred to a host, the packet is usually fragmented. The receiving systems are capable of reassembling these smaller units again to form the original data packet. On IDS, before applying any rules or try to find a signature, packet needs to be reassembled. For example, half of the signature may be present in one segment and the other half in another segment. To detect the signature correctly user has to combine all packet segments. Hackers use fragmentation to defeat intrusion detection systems.

The Detection Engine

The detection engine is the most important part of Snort. Its responsibility is to detect if any intrusion activity exists in a packet. The detection engine employs Snort rules for this purpose. The rules are read into internal data structures or chains where they are matched against all packets. If a packet matches any rule, appropriate action is taken; otherwise the packet is dropped. Appropriate actions may be logging the packet or generating alerts. The detection engine is the time-critical part of Snort. Depending upon how powerful the machine is and how many rules have been defined, it may take different amounts of time to respond to different packets. If traffic on network is too high when Snort is working in NIDS mode, some packets might be dropped and may not get a true real-time response. The load on the detection engine depends upon the following factors:

- Number of rules
- Power of the machine on which Snort is running
- Speed of internal bus used in the Snort machine
- Load on the network

The detection engine works in different ways for different versions of Snort. In all 1.x versions of Snort, the detection engine stops further processing of a packet when a rule is matched. Depending upon the rule, the detection engine takes appropriate action by logging the packet or generating an alert. This means that if a packet matches criteria defined in multiple rules, only the first rule is applied to the packet without looking for other matches. This is fine except for one problem. A low priority rule generates a low priority alert, even if a high priority rule meriting a high priority alert is located later in the rule chain. This problem is rectified in Snort version 2 where all rules are matched against a packet before generating an alert. After matching all rules, the highest priority rule is selected to generate the alert.

The detection engine in Snort version 2.0 is completely rewritten so that it is a lot faster compared to detection in earlier versions of Snort.

Logging and Alerting System

Depending upon what the detection engine finds inside a packet, the packet may be used to log the activity or generate an alert. Logs are kept in simple text files, tcpdump-style files or some other form. All of the log files are stored under /var/log/snort folder by default. User can use –l command line options to modify the location of generating logs and alerts.

Output Modules

Output modules or plug-ins can do different operations depending on how to save output generated by the logging and alerting system of Snort. Basically these modules control the type of output generated by the logging and alerting system. Depending on the configuration, output modules can do things like the following:

- Simply logging to /var/log/snort/alerts file or some other file
- Sending SNMP traps
- Sending messages to syslog facility
- Logging to a database like MySQL or Oracle.
- Generating eXtensible Markup Language (XML) output
- Modifying configuration on routers and firewalls.
- Sending Server Message Block (SMB) messages to Microsoft Windows-based machines

2.2.2 Ethereal

Simply put, Ethereal is a network analyzer. It reads packets from the network, decodes them, and presents them in an easy to understand format. Some most important aspects of Ethereal are: that it is open source, actively maintained, and free. Some important aspects of Ethereal are given below:

- It is maintained under the GNU General Public License (GPL).
- It works in promiscuous and non-promiscuous modes.
- It can capture data from the network or read from a capture file.
- It has an easy to read, and very configurable GUI.

- It has rich display filter capabilities.
- It supports Tcpdump format capture filters.

Gerald Combs first developed Ethereal in 1997 because he was expanding his knowledge of networking and needed a tool for network troubleshooting. The first version, 0.2.0, was released in July 1998.A development team, including Gilbert Ramirez, Guy Harris, and Richard Sharpe, quickly formed to provide patches, enhancements, and additional dissectors. Dissectors are what allow Ethereal to decode individual protocols and present them in readable format. Since then, a large number of individuals have contributed specific protocol dissectors that they needed and other enhancements to Ethereal. Because of the overwhelming development support and the large user base, Ethereal's capabilities and popularity continue to grow every day. [in01]

Compatibility

Ethereal can read and process capture files from a number of different products including other sniffers, routers, and network utilities. Because Ethereal uses the popular libpcap-based capture format, it interfaces easily with other products that use libpcap. It also has the capability of reading captures in a variety of other formats as well. Ethereal can automatically determine what type of file it is reading and can also uncompress gzip files. The following is the list of some products from which Ethereal can read capture files:

- Tcpdump
- Sun snoop and atmsnoop
- Microsoft Network Monitor
- Network Associates Sniffer (compressed or uncompressed) and Sniffer Pro
- Shomiti/Finisar Surveyor

Supported Protocols

When a network analyzer reads data from the network it needs to know how to interpret what it is seeing and display the output in an easy to read format. This is known as protocol decoding. Often, the number of protocols a sniffer can read and display determines its strength, thus most commercial sniffers can support several hundred protocols. Ethereal is very competitive in this area with its current support of over 480 protocols. New protocols are constantly being added by various contributors to the ethereal project. Protocol decodes, also known as dissectors, can be added directly into the code or included as plugins.

2.2.3 Honeysnap

Honeysnap is a modular, python application that can parse raw or gzipped pcap files and perform a number of diagnostics on the data. It has been designed to be easily extended to perform more diagnostic duties. It has also been designed to be minimally dependent on third party executables like tcpflow, etc.

The primary intention is to provide a first cut analysis of a directory full of pcap data, data that has probably come from a honeynet deployment using the Honeynet Project's Roo v1.x CDROM. It has the ability to decode and analyze a variety of protocols, such as HTTP [hy06], SMTP, and IRC abd can also recover files transfered. In addition it has the ability to analyze honeypot specific data sets such as SEBEK. Because of its modular nature, it is possible to add other protocols. [ho05]

Honeysnap can be run as a daily automated cron job against live honeynet data, to provide analysts with a starting point for more detailed forensic analysis. Currently the analysis performed is static, in that per run results are being stored to disk but not to a database.

Honeysnap.py is derived/inspired by work of David Watson, Steve Mumford, and Arthur Clune of the UK Honeynet Project,who wrote the first version in bash.

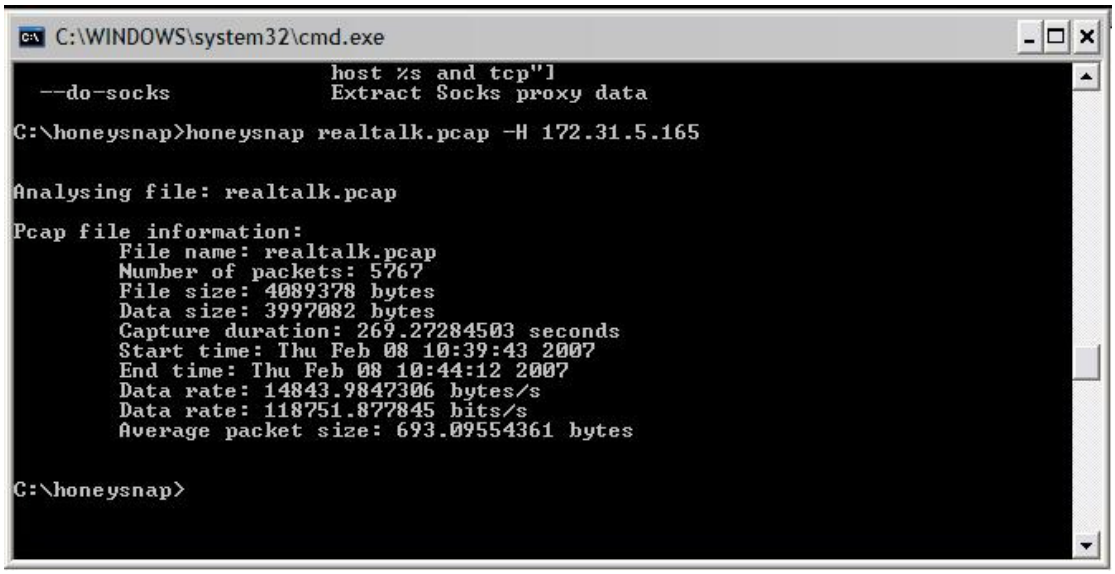


Figure 2.2: Honeysnap

An overview of what Honeysnap includes:

- Outgoing packet counts for telnet, ssh, http, https, ftp, smtp, and irc. This can be easily extended.
- Incoming and outgoing connection summaries
- Binary extraction from http, SMTP, IRC, and ftp.
- Word based inspection of IRC traffic for basic keyword profiling.
- Support for reading v2 and v3 Sebek keystroke data

Words

Honeysnap has the ability to intelligently analyze and report on specific words used in IRC communications. This is done using the 'words' file. The words file is used by Honeysnap to search IRC traffic for specific key words. User can specify a words file in Honeysnap with --words at the command line or by setting WORDFILE=/path/to/file in the configuration file. The words file should contain as words as the user requires, one per line. If no words file is provided, Honeysnap will use a built-in set of words. If a word file is provided, honeysnap will append the user-supplied list to its own list.

Honeysnap is no doubt a good tool but the only problem it gives the output in text format, i.e. Honeysnap is text based tool. All the information stored on hard disk in the form of directory and files as the output of Honeysnap. For daily analysis, it becomes typical to analyze all the data and generate some reports. Currently Honeysnap doesn't have a GUI.

Chapter 3

Problem Statement

3.1 Problem Statement

As discussed in previous chapters, there are a lot of pretty good tools available for honeypot captured data analyzing like snort, ethereal and Honeysnap.

Snort analyzes the data and returns its output in the format of alerts and logs. It does not show a full analysis of whole data. If user wants to check how much HTTP data is captured or how much SMTP data is captured, user can't. It's a good practice for a large amount of data (data of whole month, two months) but as we have intention to analyze the data every day, snort becomes the wrong choice.

Also snort uses rules to detect anomalies. All packets are checked against all rules. If rules are not written with care, they may produce false positives.

Ethereal shows the whole data in tabular form. It gives the emphasis on the protocol hierarchy. It can be useful if one needs to check some particular packets but to analyze all packets again becomes headache. In fact after finding the packets of interest, ethereal can be the best option to analyze, but in beginning there is no need to go in deep, an abstract report can do the better.

Honeysnap is a modular application, developed in python that can parse pcap files and perform a number of diagnostics on the data. It can decode and analyze a variety protocols such as HTTP [hy07], SMTP, and IRC. It can also recover files transferred in HTTP and FTP communication. It can also analyze honeypot specific data sets generated with Sebek [se01]. It means a better choice for analyzing the daily captured honeypot data.

These all information and analysis results come in text flavor, i.e. Honeysnap is text based tool. All the information stored on hard disk in the form of directory and files as

the output of Honeysnap. For daily analysis, it becomes typical to analyze all the data and generate some reports. Currently Honeysnap doesn't have a GUI.

Honeysnap produces its output in the form of files and directories. First level directory is named what honeypot is used. Second level directories are named on the name of protocol which honeysnap parsed such as HTTP, FTP, IRC and others. All connections detail is stored in the directory connections. It means if HTTP data is needed to be analyzed, whole path upto HTTP folder need to be traversed and need to check all files separately. Name of each file in HTTP folder consist of source address and port, destination address and port like 172.31.33.8:8090-172.31.33.45:7689. So if it also becomes so bulky to analyze these all data.

This thesis aims at design and development of a tool **JGSnap** that will provide a Graphical User Interface to Honeysnap, which will provide an easy to use interface to Honeysnap.

The main design objectives are:

- Parse the whole data from directory structure to mysql database.
- Generate reports that can be used to detect attacks and attackers.
- All reports must be displayed using web interface.

JGSnap is to be fully implemented with web interface. It must be developed using JSP, HTML and Java Script. To store data mysql is used.

We have discussed the idea of JGSnap with the author of Honeysnap, Mr. Arthur Clune. He was impressed with the idea and give consent for it.

3.2 Evolution of JGSnap

JGSnap is a Java-based data analyzer with a Web Graphical User Interface that provides the graphical analysis of data provided by Honeysnap. It can generate different reports for the data. One of its main characteristic is as it is being developed in Java; it can potentially run on any platform, it may be Windows, Linux or Solaris.

As database mysql [th01] is be used. Mysql is free and open source database. Only mysql connector will be supplied with JGSnap.

As JGSnap has a Web based GUI, so a Web Server is required that must support JSP and Servlets. Apache Tomcat[ap01] is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. For the implementation of JGSnap, Apache Tomcat is suitable.

Both Mysql and Apache Tomcat are available for all platform including Windows, Linux or Solaris. Some silent features of the JGSnap can be summarized as follow.

1. JGSnap will generate Trend Reports for top ten users. This Trend Report will include information such as the IP Address which accessed Honeypot most of the time, IP Address who transferred maximum data using FTP or HTTP. These Trend Reports can help to identify the attacker or the vulnerability that is mostly being used by an attacker so has the highest priority.

2. Some Graphs and Charts will be used to present the data analysis. These may include all IPv4 traffic on PIE charts. Sections of this PIE chart can be divided as TCP [rf02], UDP [rf01], ICMP, and others. Other example of graph that may include the total number of connections.

3. It will provide the summery for all connections, especially for TCP and UDP. These all information can be used to generate some reports like all connections requested by a particular user, all ports used be a particular user, connection density etc.

4. It will analyze the IRC traffic to find out total number of messages, number of unique host, number of unique commands and keywords, average number of message per host. A mouse click on host can show all the messages sent and received by the host.

5. A complete analysis report for HTTP, FTP will be generated.

6. All files transferred in HTTP and FTP communication will be listed.

Chapter 4

Proposed Architecture and Design

JGSnap is based on modular architecture. It is designed so that any new protocol and rules can be easily adapted. Its working in abstraction is as follow-

1. Fetch the data from directory structure i.e. the output of honeysnap and store it in mysql database.
2. Fetch and analyze this data with correspondent module such as TCPModule analyze TCP data and HTTP Module HTTP data.
3. Generate report on the basis of analyzed data.

Fetching the data is the most important part of the JGSnap since it is very difficult to parse data from directory structure, read them and analyze them to generate graphical output. Once data is parsed into the mysql database, it can be easily as well as more efficiently used for generating graphical output.

3.1 Architecture

Figure 3.1 shows the overall architecture of JGSnap.

Overall architecture of JGSnap is described below-

Honeysnap Output

Honeysnap produces its output in the form of directory and files. File names are decided on the basis of source IP address, source communication port and destination IP address and destination port. All files are stored in the folder named on the protocol. These folders reside in the folders named on the Honeypot.

This data structure will work as the input for the tool.

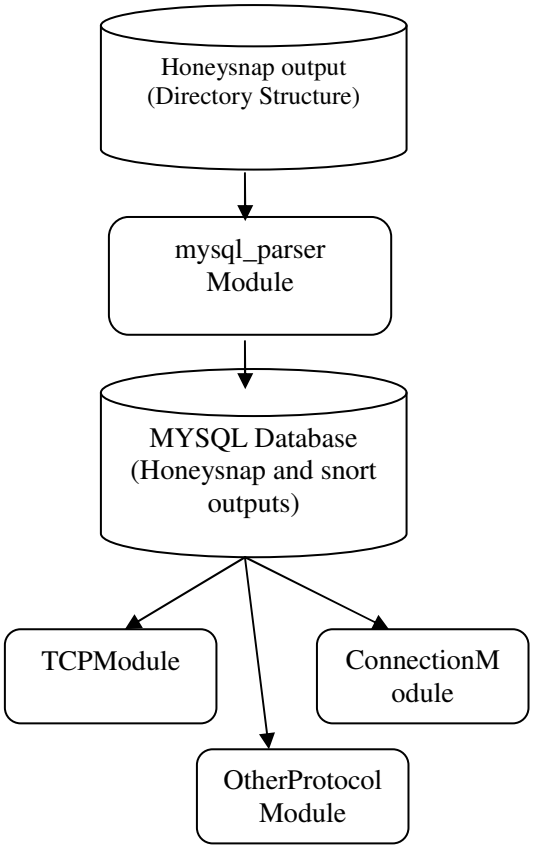


Figure 4.1: Architecture of JGSnap 1

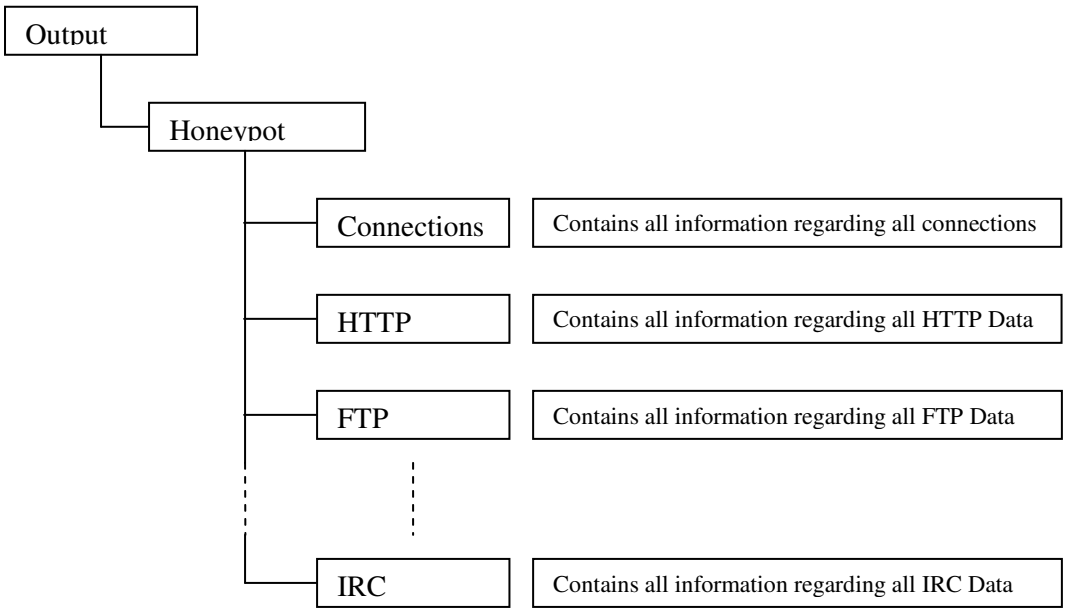


Figure 4.2: Output of Honeysnap

mysql_parser Module

mysql_parser Module is used to extract data from the directory structure i.e. output of honeysnap and stores in the mysql database. JGSnap is based on modular architecture. This module is separate from the rest of program. If any new changes are made in honeysnap data structure, only this module is required to change.

This module is designed to be robust. It can handle the errors which may be possible while reading the directory structure. Also the files in the output of honeysnap may be large (even more than 100 MB), so special care is taken while handling with these file otherwise it may crash the whole tool.

This module is designed using file handling in Java. For fast processing we used new IO package. New IO package provides better performance as compare with general java IO package.

Protocol Dependent Modules

These modules are protocol specific. Connection Module will generate report on the basis of all inbound and outbound connection. It has nothing to do with TCP and HTTP data. Similarly HTTP Module will deal with data only specific to HTTP; it doesn't need to look into other protocol's data.

If honeysnap extended for any new protocol in its specification, a module can be added to accommodate that protocol change.

These protocol dependent data is easy to analyze and also provide a basis for generating Trend Reports. If administrator want to identify which user is transferring maximum data using HTTP, can easily summarized using HTTP specific data.

3.2 Database Schema

Database schema of JGSnap is given in Figure 2. Description of the schema is given below-

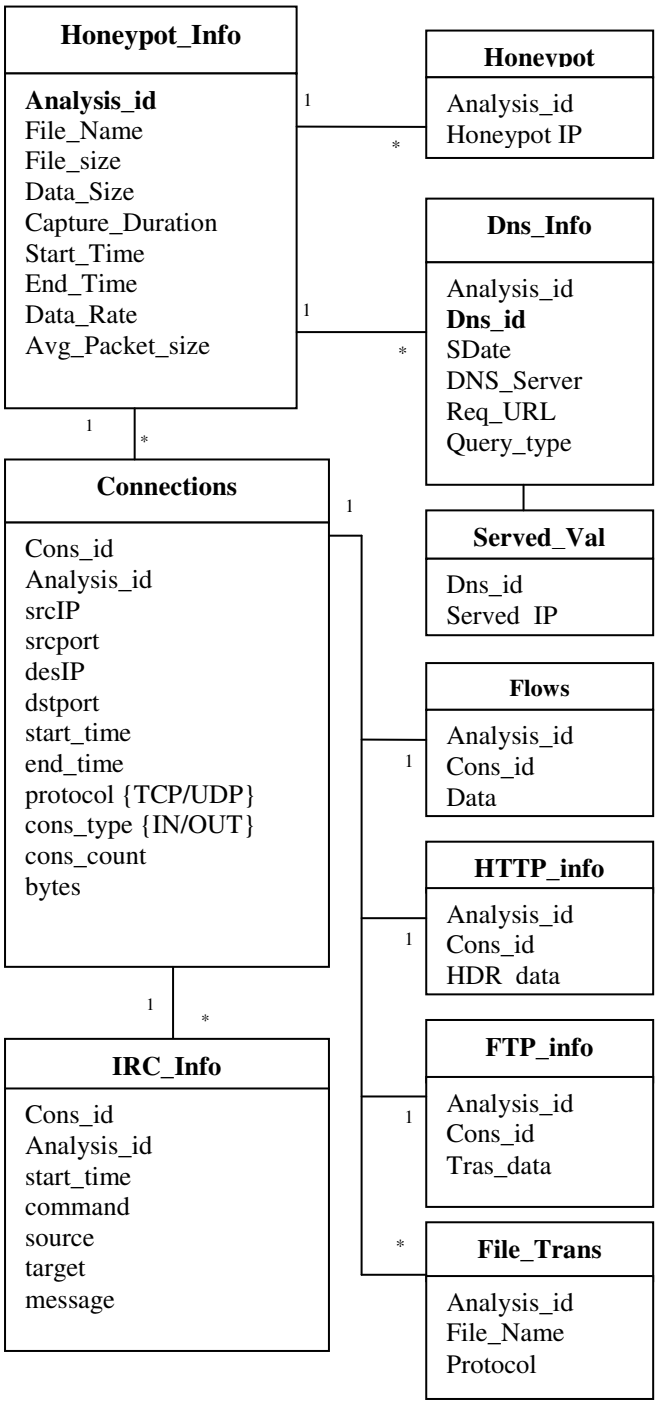


Figure 4.3: Database Schema of JGSnap

The whole database schema is supposed to implement in mysql. Some commands are supposed to be used in JGSnap that are mysql specific.

Honeypot _Info

This table will be used to store the general information about the input pcap file. It will generate the analysis_id specific for each pcap file. This analysis_id can be used to obtain information for each analysis. This table will store file name, file size, data size, captured duration, data rate and some more information.

Honeypot

This table will store the Honeypots IP addresses for each analysis_id. One analysis id may be associated with more honeypot as Honeysnap can parse more than one honeypot captured data simultaneously.

This table has 1 to * relation with honeypot_info. Analysis_id is used as the key attribute.

Dns_Info

This table will store the information about DNS queries made by Honeypot or DNS queries served by Honeypot. Each entry is given a unique ID. The need of this unique ID was that DNS server can return more than one IP for one query. We need to store all these returned IP. Served value will be store in a different table 'served_value' with dns_id.

This table has 1 to * relation with honeypot_info. Analysis_id is used as the key attribute.

Connections

This table will store the information of all connections made to or by Honeypot. This information consist source IP and port, destination IP and port, time, bytes count, connection count, protocol (TCP/UDP) and others. We shall give a cons_id to each connection. This cons_id will be used to identify connection for other protocols like HTTP, FTP.

All IP addresses in this table will be stored in decimal equivalent e.g. 172.31.33.4 will be stored as 2887721220. This step was necessary because searching, other calculation, and comparison, on integer value is much faster than string value. 172.31.33.4 must be stored as string so its decimal equivalent will be store in database.

This table has 1 to * relation with honeypot_info. Analysis_id is used as the key attribute.

Flows

Flows will store the data and Files transferred during the communication for all other protocol except HTTP and FTP.

HTTP_Info

HTTP_info will store all data transferred during HTTP communication.

FTP_Info

FTP_info will store all data transferred during FTP communication.

The difference in flows and http_info or ftp_info is that flows table data could be in binary format but http and ftp data will be in text format although these are stored in blob field. This was required so that implementation of web interface become easy.

These all tables have 1 to 1 relation with connections table. There must be one entry for one connection. This is to avoid redundancy. All the data of one connection will be stored in one field.

IRC_Info

This table will store the information about the Internet Relay Chat (IRC). It will store all commands, source, target, message information. Each record is associated with a cons_id. This cons_id is used to trace source IP and destination IP.

Chapter 5

Implementation Details & Experimental Result

Throughout the implementation of the JGSnap emphasis has been on the use of Open Source Technologies and Toolkits. The JGSnap has been implemented in JAVA programming language. JCharts API has been used for Graph based Statistical analysis, aiding in better visualization of trends. The steps followed for the implementation are described below in detail with appropriate screenshots and relevant API usage details.

Step 1: Netbeans IDE

The NetBeans IDE is a free, open-source Integrated Development Environment for software developers. The IDE runs on many platforms including Windows, Linux, Solaris, and the MacOS. It is easy to install and use straight out of the box.

The NetBeans IDE provides developers with all the tools they need to create professional cross-platform desktop, enterprise, web and mobile applications.

Netbeans provides advance source code editor. The language-aware editor indents, completes, and syntax-highlights the source code. It parses the code live, matches words and brackets, marks errors, and displays hints and javadoc. The Editor can be fully customized and split vertically or horizontally, and offers well integrated Refactoring, Debugging and JUnit testing.

Step 2: Installation of Apache Tomcat

Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process. [ap01]

Apache Tomcat is developed in an open and participatory environment and released under the Apache Software License. Apache Tomcat is intended to be a collaboration of the best-of-breed developers from around the world. These are some of the key tomcat directories, all relative to $CATALINA_HOME. CATALINA_HOME is the address of the directory where Apache Tomcat is installed.

- /bin - Startup, shutdown, and other scripts. The *.sh files (for Unix systems) are functional duplicates of the *.bat files (for Windows systems). Since the Win32 command-line lacks certain functionality, there are some additional files in here.
- /conf - Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.
- /logs - Log files are here by default.
- /webapps - This is where webapps go.

Installing Tomcat on Windows can be done easily using the Windows installer. Its interface and functionality is similar to other wizard based installers, with only a few items of interest.

- Installation as a service: Tomcat will be installed as a Windows NT/2k/XP service no matter what setting is selected. Using the checkbox on the component page sets the service as "auto" startup, so that Tomcat is automatically started when Windows starts. For optimal security, the service should be run as a separate user, with reduced permissions

- Java location: The installer will use the registry or the JAVA_HOME environment variable to determine the base path of a J2SE 5 JRE.
- Tray icon: When Tomcat is run as a service, there will be a tray icon visible. Note that when choosing to run Tomcat at the end of installation, the tray icon will be loaded at the same time.

Step 3: Installation of JCharts API

The JCharts is completely open source JAVA based API for Graph generation. The current implementation of the proposed algorithm uses JCharts [jc02] API version 0.7.5. It can be downloaded from JCharts Homepage [jc02].

The precise steps to be followed on Windows System for installation are as:

- Unzip the downloaded archive to the location intended to be used for installation.
- Copy JCharts-0.7.5.jar file in the lib folder of web application. If any jar file need to be used in web application, it should be copied in lib directory in web application. Apache tomcat automatically adds it in CLASSPATH variable.
- The JCharts API is ready to be used in JAVA applications on the Windows system.

The steps for installation on UNIX based Operating Systems are same with the exception of .tar file for JCharts Archive is to be downloaded and installed.

Step 4: Installation of mysql

MySQL is widely used on the Internet as well as on other applications. The software is available under both General Public License (GPL) and commercial licenses. The former license qualifies MySQL as legitimate open source software; but since development is for the most part done by MySQL employees, it cannot be considered the best example of an open source community. By not taking advantage of collaborative development, MySQL may evolve more slowly than it might otherwise. MySQL is best used by developers who are looking for a basic relational database that is well-proven for Web applications, low cost, and easy to learn and implement. Users who need enterprise features may consider MySQL but could find some features lacking or yet to be proven.

Installing mysql on windows is very simple with windows installer. At the time of installing, it asks for password for root, database type and some other information. Windows firewall needs to be configured to support mysql as mysql required 3306 port open for TCP communication.

Next is to create a schema[4] with name 'jgsnap' and upload the designed schema into the database.

Step 5: Development of mysql_parser Module

The whole mysql_parser is implemented in netbeans. Whole module consists of 10 classes and one example classes. Module is packaged under edu.tiet.me.abhi.parser.

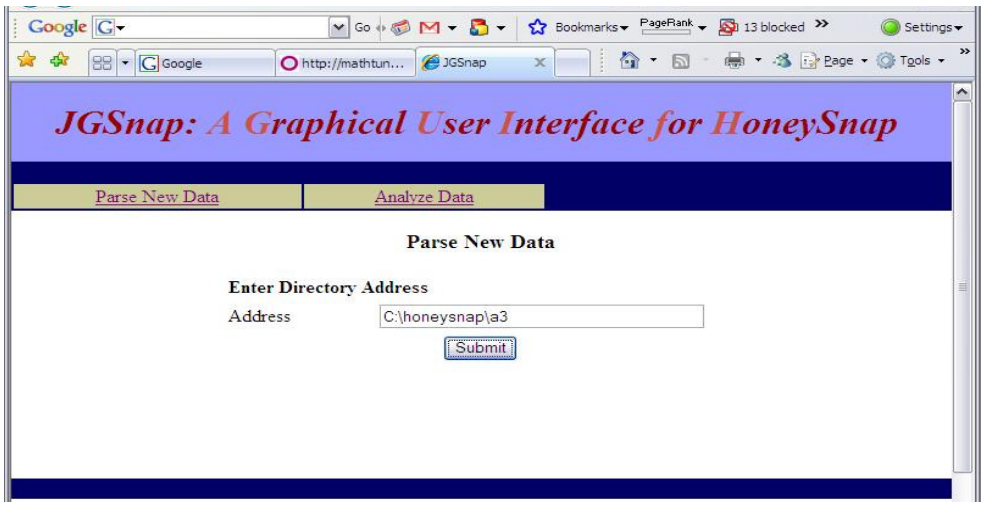
Step 6: Development of other Module

Other Modules were developed in JSP. These modules are used to visualize the output of mysql_parser module to achieve some results.

5.1 Experimental Results

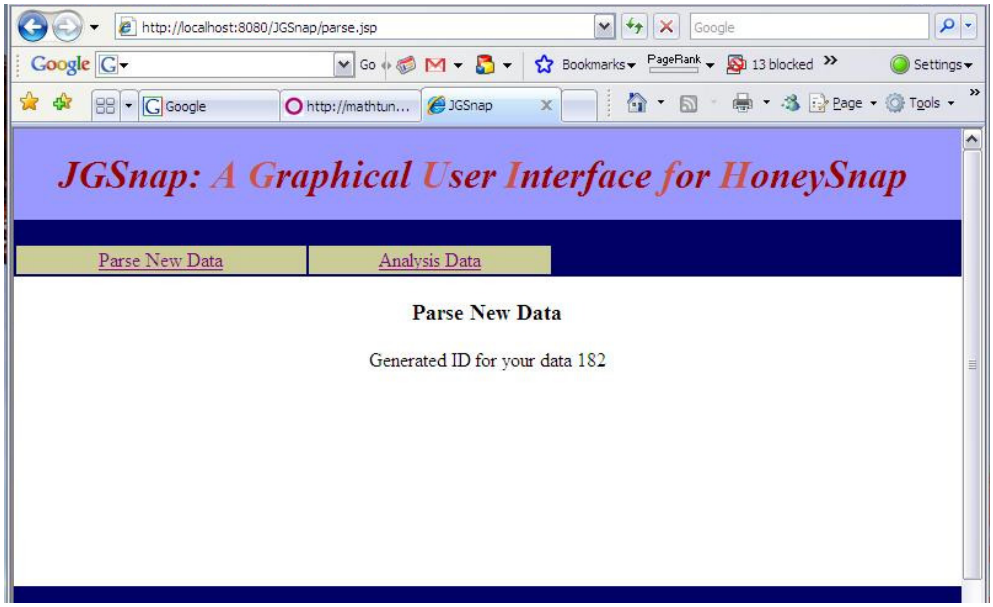
By following the steps mentioned in the last chapter, JGSnap is implemented in JAVA. In this chapter we explain and discuss the results obtained in the statistical light. For a better visualization of the trends JCharts version 0.7.5 API has been used.

5.1.2 User Input of Directory Structure



Screenshot 5.1: User Input Window

[4] Design of database schema is discussed in chapter 4.

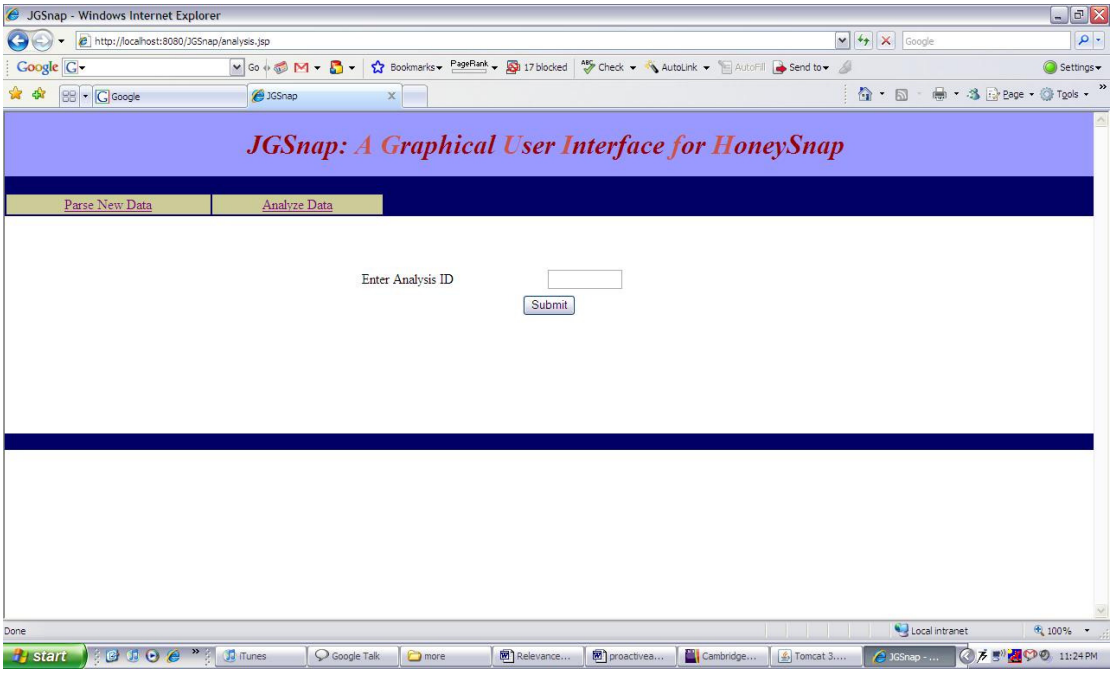


Screenshot 5.2: Generated Analysis ID 1

The above figure shows the generated ID. This ID will be used to fetch and generate report from the given data. For the example this ID is 182.

5.1.2 Analysis of parsed Data

The following Figure shows the analysis part of JGSnap. This part is provided with Web Interface. The following form is used to enter analysis id as input.

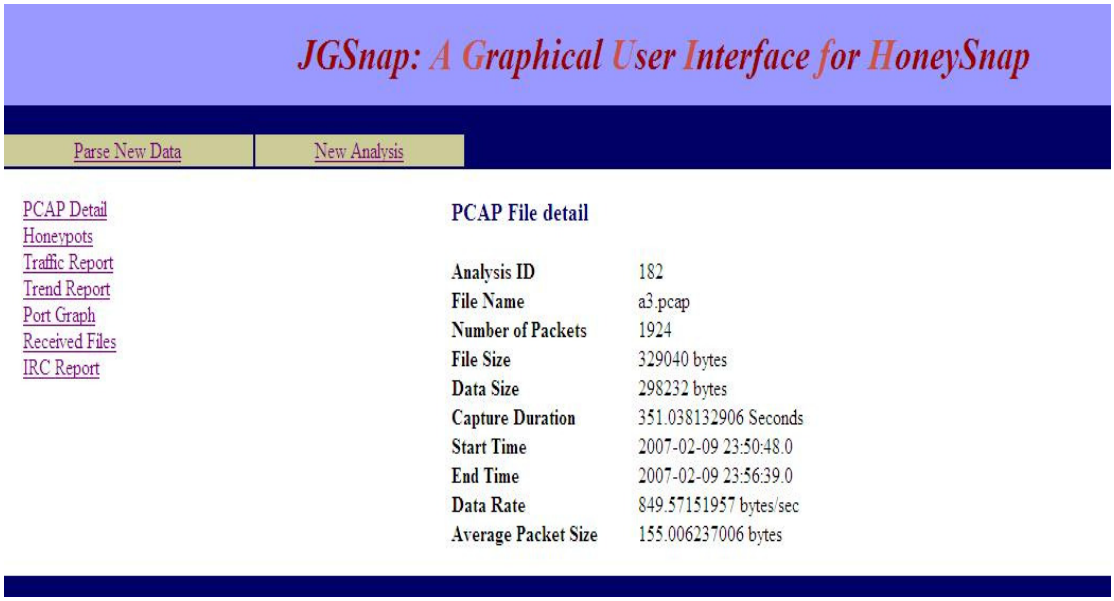


Screenshot 5.3: Analysis ID Input Form

After submitting the id, the following screen will appear. This report shows the basic information about the pcap file such as File Name, Number of Packets, File Size. This page and all other have a navigation panel on the left. This navigation panel consist the links to other forms. These forms are including the current:

1. PCAP Detail
2. Honeypots
3. Traffic Report
4. Trend Report
5. Port Graph
6. Received Files
7. IRC Report

1. PCAP Detail:

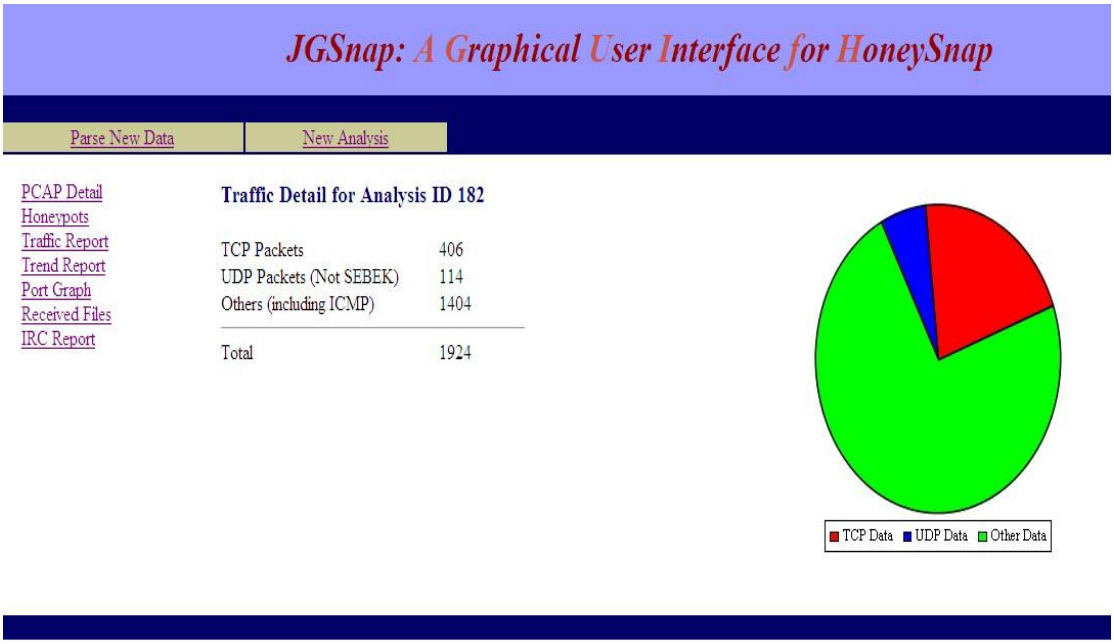


Screenshot 5.4: PCAP Detail Form

2. **Honeypot Report**: The following report shows the honeypot(s) used in the parsing of pcap file by Honeysnap.

Screenshot 5.5: Honeypot Form

3. **Traffic Report**: Traffic report summarizes all the traffic detail. It shows count of TCP, UDP and other packets. Other packet also includes ICMP packets. In UDP, packets, which are coming to port 1101, are not counted. These packets are meant for SEBEK. These detail are also shown in pie chart. Legends are
    a. RED : TCP Packets
    b. BLUE: UDP Packets
    c. GREEN: Other Packets



Screenshot 5.6: Traffic Report

4. **Trend Report:** Trend Report is used to show top ten users for each protocol i.e. UDP and TCP. Each Protocol section has two sub reports. First one is used to show the top ten users that transferred highest bytes. Second one is used to show the list of top ten users who made maximum number of connections.

   Same trend reports are shown for UDP also.

   Following is the report showing top ten users that transferred maximum bytes. First column shows the IP address of source machine and port number. Second column shows the destination address with port number. Third column and fourth column respectively shows start and end time of the connections. Next one is connection count and byte transferred.

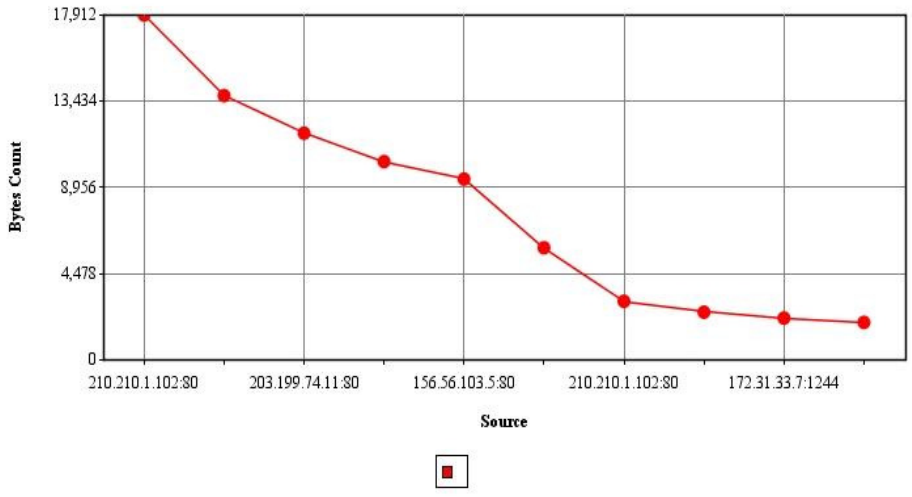
   Report is sorted on the basis of bytes transferred.

TCP

Top 10 users for bytes transfer

Source	Destination	Start Time	End Time	Cons Count	Bytes Transferred
210.210.1.102:80	172.31.33.7:1260	2007-02-09 18:25:16.0	2007-02-09 18:25:58.0	18	17907
203.199.74.11:80	172.31.33.7:1264	2007-02-09 18:25:50.0	2007-02-09 18:26:24.0	16	13720
203.199.74.11:80	172.31.33.7:1255	2007-02-09 18:24:45.0	2007-02-09 18:25:20.0	15	11779
210.210.1.102:80	172.31.33.7:1254	2007-02-09 18:24:40.0	2007-02-09 18:25:16.0	12	10302
156.56.103.5:80	172.31.33.7:1261	2007-02-09 18:25:28.0	2007-02-09 18:26:07.0	12	9396
72.14.253.95:80	172.31.33.7:1253	2007-02-09 18:24:33.0	2007-02-09 18:25:08.0	9	5824
210.210.1.102:80	172.31.33.7:1268	2007-02-09 18:25:58.0	2007-02-09 18:26:26.0	7	3049
64.233.163.111:465	172.31.33.7:1245	2007-02-09 18:21:35.0	2007-02-09 18:22:43.0	23	2490
172.31.33.7:1244	64.233.163.111:465	2007-02-09 18:20:50.0	2007-02-09 18:21:05.0	7	2177
172.31.33.7:1245	64.233.163.111:465	2007-02-09 18:21:35.0	2007-02-09 18:22:42.0	21	1956

Screenshot 5.7: Trend Report TCP



Screenshot 5.8: Trend Report Graph TCP 1

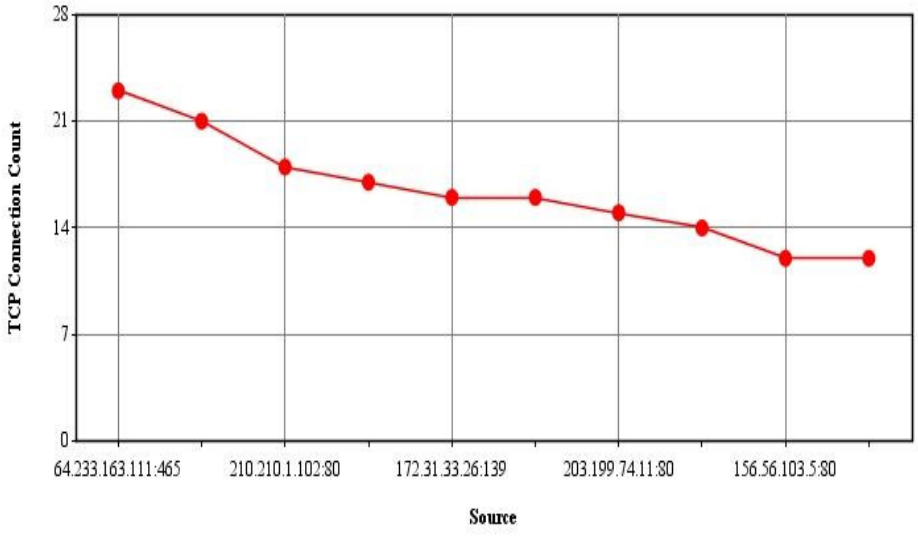
Following report shows the top ten users who made maximum connections. This table is sorted on the basis of connection count.

Top 10 users for connection count

Source	Destination	Start Time	End Time	Cons Count	Bytes Transferred
64.233.163.111:465	172.31.33.7:1245	2007-02-09 18:21:35.0	2007-02-09 18:22:43.0	23	2490
172.31.33.7:1245	64.233.163.111:465	2007-02-09 18:21:35.0	2007-02-09 18:22:42.0	21	1956
210.210.1.102:80	172.31.33.7:1260	2007-02-09 18:25:16.0	2007-02-09 18:25:58.0	18	17907
172.31.33.7:1250	172.31.33.26:139	2007-02-09 18:23:56.0	2007-02-09 18:23:56.0	17	1729
172.31.33.26:139	172.31.33.7:1250	2007-02-09 18:23:56.0	2007-02-09 18:23:56.0	16	1732
203.199.74.11:80	172.31.33.7:1264	2007-02-09 18:25:50.0	2007-02-09 18:26:24.0	16	13720
203.199.74.11:80	172.31.33.7:1255	2007-02-09 18:24:45.0	2007-02-09 18:25:20.0	15	11779
172.31.33.7:1260	210.210.1.102:80	2007-02-09 18:25:16.0	2007-02-09 18:25:58.0	14	712
156.56.103.5:80	172.31.33.7:1261	2007-02-09 18:25:28.0	2007-02-09 18:26:07.0	12	9396
172.31.33.7:1264	203.199.74.11:80	2007-02-09 18:25:47.0	2007-02-09 18:26:24.0	12	520

Screenshot 5.9: Trend Report Graph TCP Connection

Following graph visualize the above table in graphical form. On the X Axis source IP address are taken and on Y Axis connection count is taken.



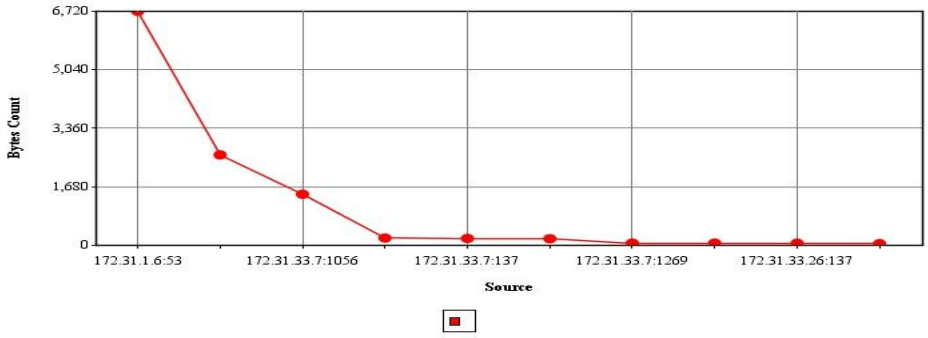
Screenshot 5.10: Trend Report Graph TCP Connection Graph

The following are the UDP reports similar to TCP as discussed above.

UDP

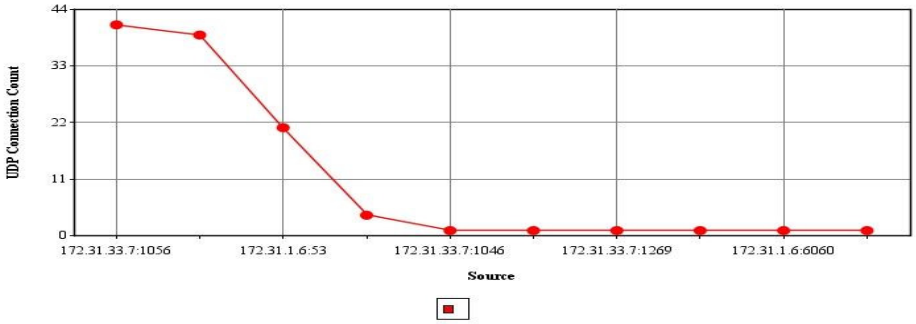
Top 10 users for bytes transfer

Source	Destination	Start Time	End Time	Cons Count	Bytes Transferred
172.31.1.6:53	172.31.33.7:1056	2007-02-09 18:23:07.0	2007-02-09 18:26:35.0	21	6717
172.31.33.7:137	172.31.1.4:137	2007-02-09 18:20:56.0	2007-02-09 18:25:12.0	39	2598
172.31.33.7:1056	172.31.1.6:53	2007-02-09 18:23:03.0	2007-02-09 18:26:28.0	41	1467
172.31.1.6:53	172.31.33.7:1046	2007-02-09 18:26:26.0	2007-02-09 18:26:26.0	1	212
172.31.33.7:137	172.31.33.255:137	2007-02-09 18:23:56.0	2007-02-09 18:24:00.0	4	200
172.31.33.7:137	172.31.33.90:137	2007-02-09 18:22:20.0	2007-02-09 18:22:20.0	1	193
172.31.33.7:1269	172.31.1.6:6060	2007-02-09 18:26:04.0	2007-02-09 18:26:04.0	1	65
172.31.33.7:1246	172.31.1.6:6060	2007-02-09 18:23:04.0	2007-02-09 18:23:04.0	1	65
172.31.33.26:137	172.31.33.7:137	2007-02-09 18:23:56.0	2007-02-09 18:23:56.0	1	62
172.31.33.90:137	172.31.33.7:137	2007-02-09 18:22:20.0	2007-02-09 18:22:20.0	1	50



Top 10 users for connection count

Source	Destination	Start Time	End Time	Cons Count	Bytes Transferred
172.31.33.7:1056	172.31.1.6:53	2007-02-09 18:23:03.0	2007-02-09 18:26:28.0	41	1467
172.31.33.7:137	172.31.1.4:137	2007-02-09 18:20:56.0	2007-02-09 18:25:12.0	39	2598
172.31.1.6:53	172.31.33.7:1056	2007-02-09 18:23:07.0	2007-02-09 18:26:35.0	21	6717
172.31.33.7:137	172.31.33.255:137	2007-02-09 18:23:56.0	2007-02-09 18:24:00.0	4	200
172.31.33.7:1046	172.31.1.6:53	2007-02-09 18:26:26.0	2007-02-09 18:26:26.0	1	34
172.31.33.7:1246	172.31.1.6:6060	2007-02-09 18:23:04.0	2007-02-09 18:23:04.0	1	65
172.31.33.7:1269	172.31.1.6:6060	2007-02-09 18:26:04.0	2007-02-09 18:26:04.0	1	65
172.31.33.7:137	172.31.33.90:137	2007-02-09 18:22:20.0	2007-02-09 18:22:20.0	1	193
172.31.1.6:6060	172.31.33.7:1246	2007-02-09 18:23:04.0	2007-02-09 18:23:04.0	1	5
172.31.1.6:53	172.31.33.7:1046	2007-02-09 18:26:26.0	2007-02-09 18:26:26.0	1	212



Screenshot 5.11: Trend Report Graph UDP

5. **Port Graph**: The following are the contents are Port Graph report. Here all the connections which had no bytes transferred but made connections are shown. These sources are vulnerable as they might be scanning the port. This report shows top ten users.

   A graph is also used to visualize the contents of the table. X Axis shows the source IP and Y Axis shows the Connection count.

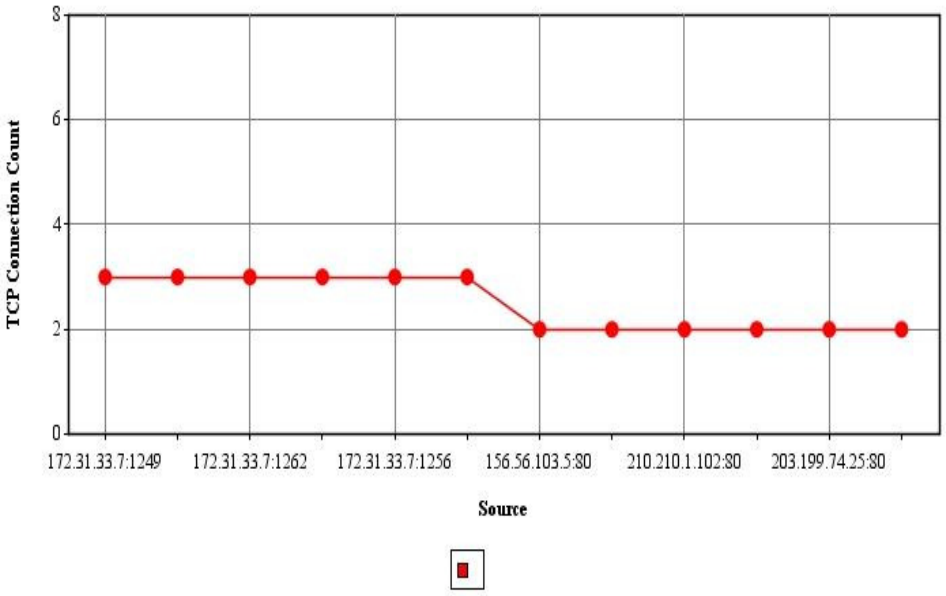
   Same reports are used for UDP protocol.

TCP Port Scanning

Following are the ports that has byte transfer = 0, i.e. that ports were tried to scan but that port may be closed.

Source	Destination	Start Time	End Time	Cons Count
172.31.33.7:1249	203.199.74.11:80	2007-02-09 18:23:36.0	2007-02-09 18:23:45.0	3
172.31.33.7:1267	66.249.81.147:80	2007-02-09 18:25:56.0	2007-02-09 18:26:05.0	3
172.31.33.7:1262	64.233.189.104:80	2007-02-09 18:25:30.0	2007-02-09 18:25:39.0	3
172.31.33.7:1265	64.233.189.104:80	2007-02-09 18:25:51.0	2007-02-09 18:26:00.0	3
172.31.33.7:1256	156.56.103.5:80	2007-02-09 18:25:05.0	2007-02-09 18:25:13.0	3
172.31.33.7:1248	202.164.55.115:80	2007-02-09 18:23:22.0	2007-02-09 18:23:31.0	3
156.56.103.5:80	172.31.33.7:1272	2007-02-09 18:26:21.0	2007-02-09 18:26:21.0	2
156.56.103.5:80	172.31.33.7:1273	2007-02-09 18:26:21.0	2007-02-09 18:26:21.0	2
210.210.1.102:80	172.31.33.7:1274	2007-02-09 18:26:35.0	2007-02-09 18:26:35.0	2
66.249.81.104:80	172.31.33.7:1271	2007-02-09 18:26:20.0	2007-02-09 18:26:20.0	2
203.199.74.25:80	172.31.33.7:1275	2007-02-09 18:26:28.0	2007-02-09 18:26:28.0	2
64.233.189.104:80	172.31.33.7:1276	2007-02-09 18:26:36.0	2007-02-09 18:26:36.0	2

Screenshot 5.12: TCP Port Analysis

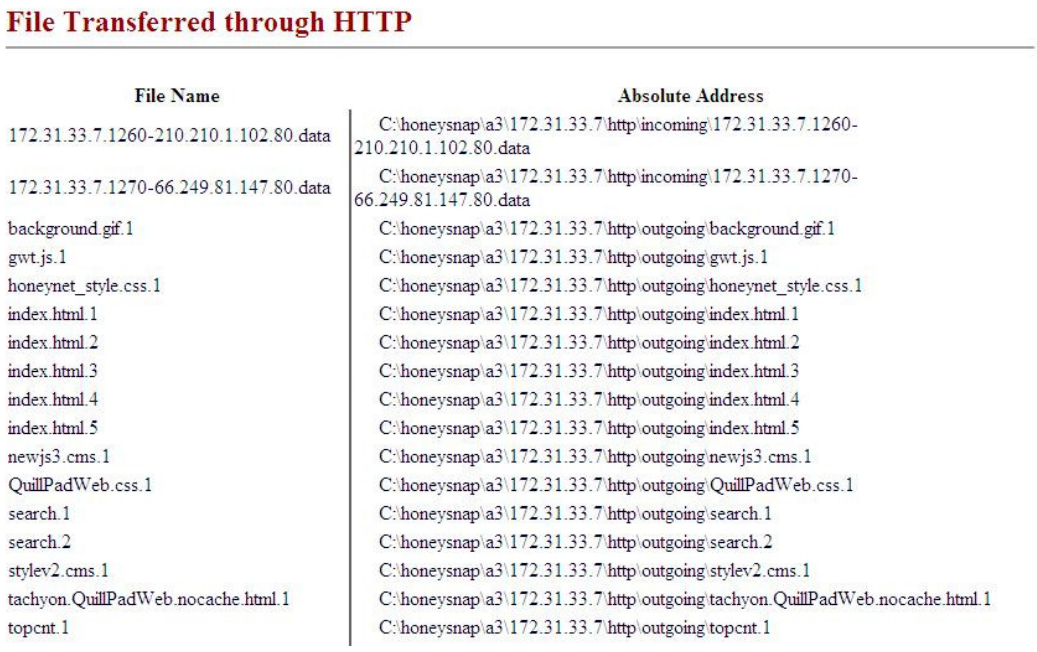


Screenshot 5.13: TCP Port Graph

6. **Received Files**: This report shows list of all the files that were received during HTTP or FTP communication.

The following table shows all the files transferred durinh HTTP communication. All data is sorted on the file name.

First column shows the name of all files transferred while second column shows absolute address of the file. Contents of the file can be checked from the absolute address.



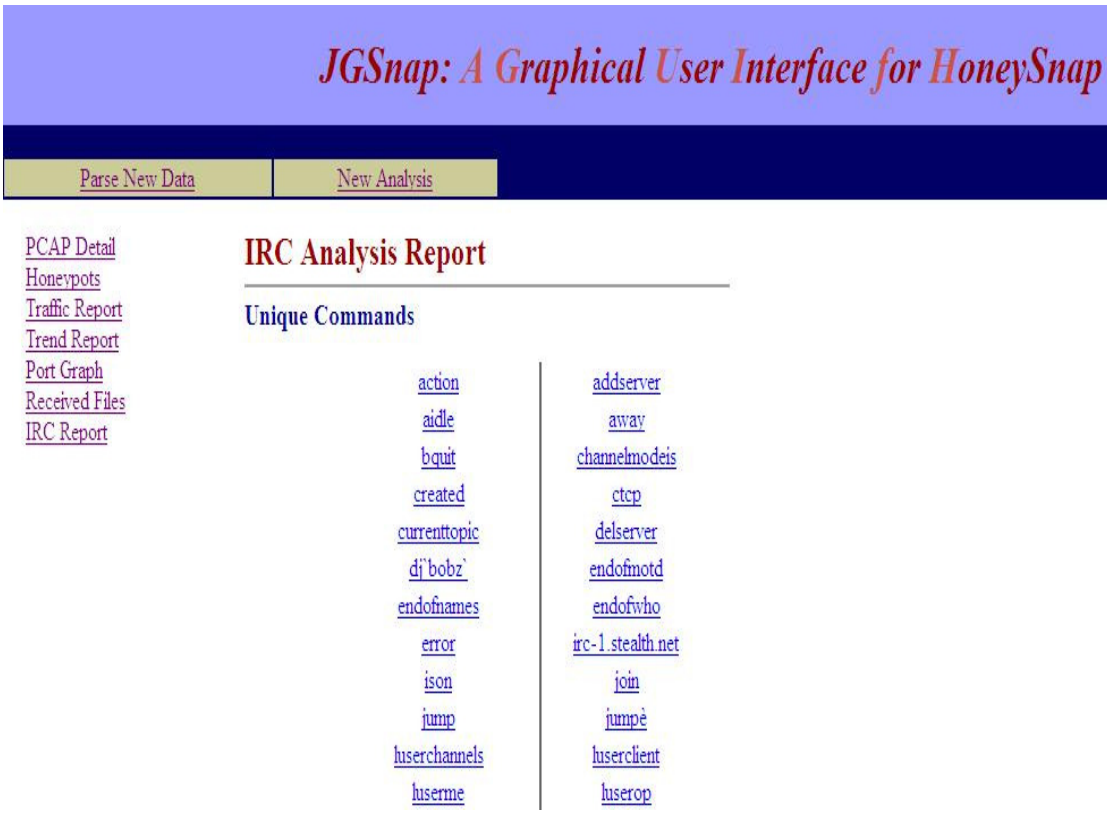
File Transferred through HTTP

File Name	Absolute Address
172.31.33.7.1260-210.210.1.102.80.data	C:\honeysnap\a3\172.31.33.7\http\incoming\172.31.33.7.1260-210.210.1.102.80.data
172.31.33.7.1270-66.249.81.147.80.data	C:\honeysnap\a3\172.31.33.7\http\incoming\172.31.33.7.1270-66.249.81.147.80.data
background.gif.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\background.gif.1
gwt.js.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\gwt.js.1
honeynet_style.css.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\honeynet_style.css.1
index.html.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\index.html.1
index.html.2	C:\honeysnap\a3\172.31.33.7\http\outgoing\index.html.2
index.html.3	C:\honeysnap\a3\172.31.33.7\http\outgoing\index.html.3
index.html.4	C:\honeysnap\a3\172.31.33.7\http\outgoing\index.html.4
index.html.5	C:\honeysnap\a3\172.31.33.7\http\outgoing\index.html.5
newjs3.cms.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\newjs3.cms.1
QuillPadWeb.css.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\QuillPadWeb.css.1
search.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\search.1
search.2	C:\honeysnap\a3\172.31.33.7\http\outgoing\search.2
stylev2.cms.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\stylev2.cms.1
tachyon.QuillPadWeb.nocache.html.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\tachyon.QuillPadWeb.nocache.html.1
topcnt.1	C:\honeysnap\a3\172.31.33.7\http\outgoing\topcnt.1

Screenshot 5.14: Received Files

7. **IRC Report:** IRC stand for Internet Relay Chat. IRC is very popular on Internet. Internet Relay Chat (IRC) is a form of real-time Internet chat or synchronous conferencing. It is mainly designed for group (many-to-many) communication in discussion forums called channels, but also allows one-to-one communication and data transfers via private message.

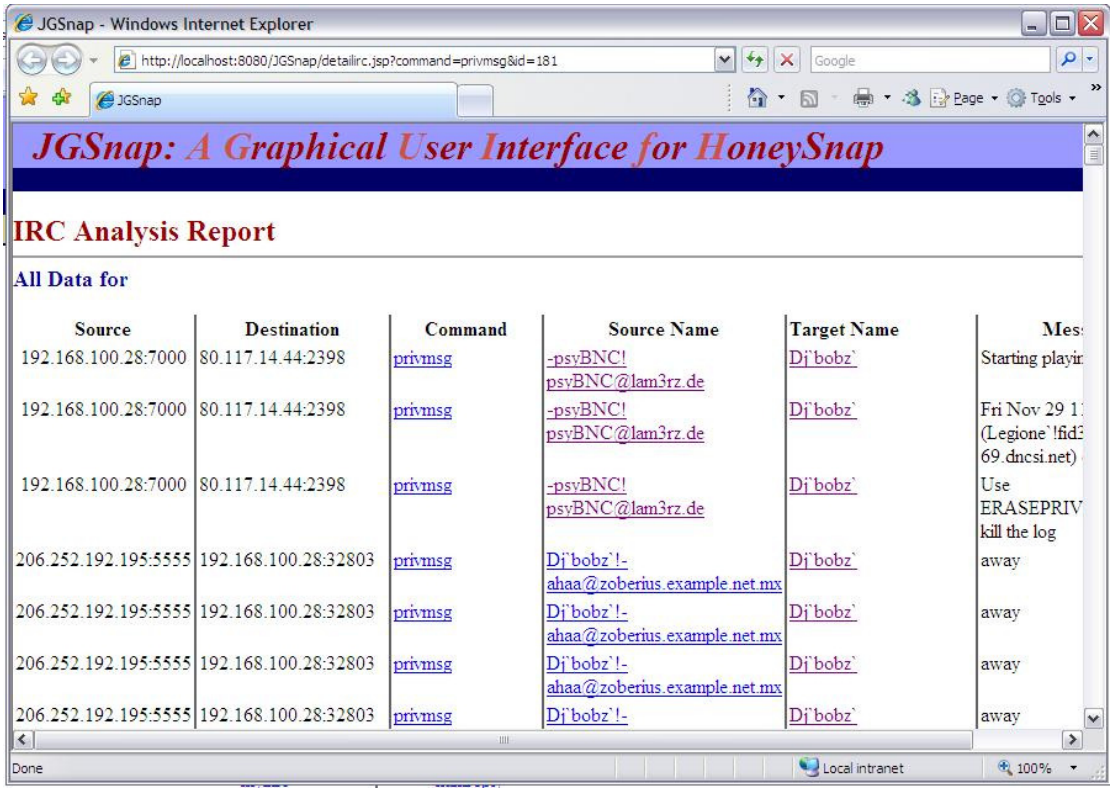
In the following form all the commands used in IRC chat are shown below. On clicking on any of the command listed below, it will open a new pop up window that will have all the detail of messages for the particular command.



Screenshot 5.15: IRC Analysis Report

Above report shows all the commands used in the communication. These commands are sorted in ascending order.

The following form shows the detail for "privmsg" command. If user wants to see the messages for a particular source, click on the source name. Similarly if user wants to see the messages for a particular target, click on it. It will show all the messages which have the same target name.



Screenshot 5.16: IRC Analysis Report for Specific Command

Conclusions and Future Scope

The work presented in this thesis provides an insight into the world of Internet, Network Security, Honeypots, and Data analyzer tool such as Snort, Ethereal, and Honeysnap.

Honeypots are becoming very popular in the area of Network Security. A honeypot is a closely monitored computing resource that intended to be probed, attacked, or compromised. The value of a honeypot is determined by the information that can be obtained from it. This information is further processed to achieve some kind of information about attacks and attacker.

There are a lot pretty good tools are available for analyzing the data logged by Honeypot such as Snort, Ethereal, and Honeysnap.

Snort analyzes the data and returns its output in the format of alerts and logs. It does not show a full analysis of whole data. Ethereal gives the emphasis on the protocol hierarchy. It can be useful if we need to check some particular packets but to analyze all packets again becomes headache. Honeysnap provides all information and analysis results in text flavor, i.e. Honeysnap is text based tool. All the information stored on hard disk in the form of directory and files as the output of Honeysnap. For daily analysis, it becomes typical to analyze all the data and generate some reports. Currently Honeysnap doesn't have a GUI.

The thesis work focuses primarily on JGSnap. JGSnap is a Java-based data analyzer with a Web Graphical User Interface that provides the graphical analysis of data provided by Honeysnap. It can generate different reports for the data. One of its main characteristic is as it is developed in Java; it can potentially run on any platform, it may be Windows, Linux or Solaris.

1. JGSnap is able to generate Trend Reports for top ten users.

2. Some Graphs and Charts are used to present the data analysis. These include all IPv4 traffic on PIE charts. Sections of this PIE chart can be divided as TCP, UDP [rf01], ICMP, and others. Other example of graph that may include the total number of connections.

3. It will provide the summery for all connections, especially for TCP and UDP. These all information can be used to generate some reports like all connections requested by a particular user, all ports used be a particular user, connection density etc.

4. It analyzes the IRC traffic to find out total number of messages, number of unique commands and keywords. A mouse click on host can show all the messages sent and received by the host.

5. A complete analysis report for HTTP, FTP will be generated.

6. All files transferred in HTTP and FTP communication will be listed.

7. Complete tool is designed using open source tools and techniques, which include Java, Apache Tomcat, mysql and some others.

Future Scope

- Currently this tool is taking Honeysnaps output as input and further after processing it insert the whole information into mysql database. In future a script can be added to honeysnap, so that it directly enters the information into the mysql. It will boost the efficiency and speed.

- A report can be added to search a particular host. It can give the whole statistics about the host.

- Some more reports including DNS report, HTTP data analysis, can be added.

References

[ap01] Apache Tomcat, http://tomcat.apache.org/

[ba01] Back Officer Friendly, http://www.nfr.com/resource/backOfficer.php

[ce01] CERT, CSIRT Services © 2002 Carnegie Mellon University; Stelvio, The Netherlands; PRESECURE Consulting GmbH, Germany.
< www.cert.org/archive/pdf/CSIRT-services-list.pdf>

[co02] Cohen, F., *Protection by Deception*, in *Network Security Magazine – Managing network security*. September 2002. p. 17-19.

[ga01] Gary Miliefsky, A Guide to Proactive Network Security, Published on ZDNet News: November 30, 2004, 5:41 AM PT

[ho01] Honeypot Software, Honeypot Products, Deception Software, http://www.honeypots.net/honeypots/products

[ho02] Honeyd, http://www.citi.umich.edu/u/provos/honeyd/

[ho03] Honeywall, http://www.honeynet.org/tools/cdrom/

[ho04]HoneyBot, http://www.atomicsoftwaresolutions.com/honeybot.php

[ho05] Honeysnap Homepage
http://honeynet.tiet.ac.in/tools/honeysnap/index.html

[hy06] Hypertext Transfer Protocol -- HTTP/1.1
www.w3.org/Protocols/rfc2616/rfc2616.html

[in01] Introducing Ethereal: Network Protocol Analyzer,

http://ethereal.com/docs/syngress-book/284_EPS_02.pdf

[in02] Intrusion Detection Systems with Snort, http://www.phptr.com/content/images/0131407333/downloads/0131407333.pdf

[in03] Internet Usage World Status, http://www.internetworldstats.com/

[ja01] Jacco Tunnissen, "Honeypots, Intrusion Detection, Incident Response", http://www.honeypots.net

[jc02] JCharts, http://jcharts.sourceforge.net/

[ko01] Kossakowski, Klaus-Peter, Information Technology Incident Response Capabilities. Hamburg: Books on Demand, 2001 (ISBN: 3-8311-0059-4).

[kf02] KFSensor, http://www.keyfocus.net/kfsensor/

[la01] Lance Spitzner, "Fighting Relay Spam the Honeypot Way", http://www.trackinghackers.com/solutions/sendmail.html

[lo01] Lawrence R. Rogers, Home Computer and Internet User Security, CERT® Training and Education, © 2005 Carnegie Mellon University

[ni01] Niels Provos, A Virtual Honeypot Framework, Google, Inc.

[pr01] Proxypot, http://www.proxypot.org/

[pa02] Paul A. Taylor, Hackers: Crime in the Digital Sublime. Routledge, 1999.

[se01] Sebek™ Homepage http://honeynet.tiet.ac.in/tools/sebek/

[se02] Secure Computing Corporation, Intrusion Prevention System, Part I Deciphering the inline Intrusion Prevention hype, and working toward a real-

world, proactive security solution, www.securecomputing.com, ©2003 Secure Computing Corporation. All Rights Reserved.

[se03] Sebastian Wolfgarten, Honeypots in a Nutshell

[se04] Spitzner, L., *Definition and value of Honeypots*, in *Tracking Hackers*. 2003. http://www.trackinghackers.com/papers/honeypots.html

[th01] The world's most popular open source database http://www.mysql.com/

[rf01] RFC 768 (rfc768) - User Datagram Protocol www.faqs.org/rfcs/rfc768.html

[rf02] RFC 793 (rfc793) -Transmission Control Protocol www.faqs.org/rfcs/rfc793.html

[wi01] William Stalling, Cryptography and Network Security Principles and Practices, Third Edition, Pearson Education, 2002

List of Publications

1. Abhishek Vershney, Maninder Singh, "*JGSnap: A Graphical Analyzer for Honeysnap* ", All India Seminar on Cyber Crime and Security, Organized by Institution of the Engineers, Lucknow, (13-15 April)