**International Journal of Computer Science and Engineering (IJCSE)**
**ISSN 2278-9960**
**Vol. 2, Issue 4, Sep 2013, 27-36**
**© IASET**

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

# IMPLEMENTATION OF ATTACK DATA COLLECTION INCORPORATING MULTI LEVEL DETECTION CAPABILITIES USING LOW INTERACTION HONEYPOT

**RUBY KAMBOJ & VANITA RANA**

Department of Computer Science, Indo Global College of Engineering, Punjab Technical University
Begowal, Punjab, India

## ABSTRACT

Nowadays, security system is very important to any organization to protect their data or any information kept in their computer from the intruders to access. A honeypot is a type of information system that is used to obtain information on intruders in a network. This paper proposes a methodology for establishing a virtual Honeypot on a Virtualbox Server running dionaea. The implementation is specific to a Linux based host having a single physical network interface card. An effort has been made to ensure that all the software (both the OS and associated tools) used for the project are either free or Open Source. Special techniques were implemented in order to enhance the data capture mechanisms on the Linux-based Honeypot to efficiently generate reports. The ultimate goal of Attack Data Collection system is to detect and identify any malicious activity coming from the Internet. This system incorporates multi level detection by using vulnerabilities based attack data collection and network intrusion detection based attack data collection. Our system is tested by visiting of various malicious websites and detection of malwares dropped on the system is detected and logged in the system database.

**KEYWORDS:** Study of Network Security, Interaction, Honeypot, Attack, Level

## INTRODUCTION

Within last couple of decades, industry as well as governments uses Internet at an increasing pace in basic functions and core activities. Governments use Internet to provide citizens and businesses with public services. The revolution in Information Technology has provided a flood of assets in the form of applications and services. Enterprises have based their entire business models on top of these assets. Networks have evolved from low speed half duplex links to full duplex, multi-homed, self convergent, gigabyte streams, controlled by advanced protocols.

The security of the available applications and services accessible over these networks currently represents a major challenge to the IT industry. Each day, exploits, worms, viruses and buffer overflows severely threaten the IT infrastructure and associated business assets along with mission critical systems. By learning the tactics and techniques used by malicious black hats crackers, we can secure our IT assets and infrastructure. Honeypots provide a means to study black-hat techniques and tactics through which they have been able to gain illegitimate access to system resources along with methods for analyzing the tools that they use to obtain this access. This is achieved by setting up a vulnerable environment that poses as a valid resource to any attacker, but is heavily logged.

Most network security tools are passive in nature; for example, firewalls and IDS. They operate on available rules and signatures in their database. Anomaly detection is limited only to these set of available rules. Any activity not in alignment with those rules goes undetected. Honeypots by design allow you to take the initiative by turning the tables on malicious black hats. The Honeypot system has no production value and has no authorized activity. Thus any interaction with the Honeypot is most likely the result of malicious intent. Honeypots do not solve the security problem but provide data and knowledge that aids the system administrator in enhancing the overall security of their network.

There are many advantages of using honeypots over other network security tools like intrusion detection system. As Honeypot collects data that is related only to attack or unauthorized activity, the data sets are very small but they carry large amount of information in them. The attacker's activities are captured in detail. Unlike most of the intrusion detection/prevention systems, honeypots produce a negligible percentage of false positives. It logs everything that comes there way including all the new tools and techniques that the attackers use. Honeypot are quite capable of working in the encryption enabled environment. Also, it is IPv6 enabled. With the scarcity of IP in today's world, shifting to IPv6 is unavoidable. Hence honeypot will also continue to work effectively in future. It is very simple to understand, deploy and maintain. It does not require costly resources and can be setup using cheap systems[2]. This knowledge can be used as input to any early warning systems. Over the years, researchers have successfully isolated and identified worms and exploits using Honeypots placed in specialized architectures called Honeynets. These are then used for signature and rule development. Honeynets are capable of logging far more information than any other available security tools. They give insight into attacks and attackers, their skill level, their organization as groups or individuals, their motives and tactics; and thus, almost every aspect is logged and can be made auditable [4]. Virtualization technologies like Virtualbox provide the ability and flexibility to create a specialized network of hosts on a single physical machine. This has considerably reduced hardware costs.

This project has been setup using free and Open Source tools and technologies that run on a Linux platform. Linux operating systems have been used as the host OS and for guest OS virtual machines. This includes a Linux based Honeypot and a dionaea gateway. Based on the results of this study we can enhance the overall security of our network resources. We look at the objectives behind the deployment of honeypots and their uses. We summarize by presenting a survey of existing honeypot technologies. In this thesis we look at the new concept of honeypots and their application in intrusion detection systems. As a part of the thesis project a honeypot was designed and implemented. The honeypot was kept online for a period of time and any network communication or events related to it was recorded and analyzed. The remainder of the paper is organised as follows: Section II describes the background and the technology that has been employed. Section III discusses the tools used for setting up the honeypot resource. Section IV describes the proposed architecture of the project. Experimentation results are discussed in Section V. Section VI summarizes the project. Section VII discusses the conclusion and future work to enhance this technology.

## BACKGROUND

### Honeypots

A Honeypot is generally defined as a network security resource whose value lies in it being scanned, attacked, compromised, controlled and misused by an attacker to achieve his malicious goals. Lance Spitzner defines Honeypots as "A Honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource" [4].

Honeypots can be classified into two main categories. Firstly, they can be based upon their level of interaction with an attacker. This can be further categorized as:

- **Low-Interaction**: Emulate a variety of host services. These mimic real services but are implemented as a sandbox environment and run as an application. e.g. honeyd and nepenthes.
- **High-Interaction**: Attacker is given the freedom to interact with a real operating system and their every attempt is logged and accounted for.

The second Honeypot category is identified by the way they are deployed in a network. This includes:

- **Production Honeypots**: They are placed within an organization's production network for the purpose of detection. They extend the capabilities of intrusion detection systems. Such Honeypots are developed and configured to integrate with the organization's infrastructure. They are usually implemented as low-interaction Honeypots sitting within the server farm, but implementations may vary depending on available funding and requirements of the organization.

- **Research Honeypots**: These are deployed by network security researchers – the white hat hackers. They allow complete freedom for the attacker and, in the process; it is possible to learn their tactics. Using Research Honeypots zero day exploits, Worms, Trojans and viruses propagating in the network can be isolated and studied. Researchers can then document their findings and share them with system programmers, network and system administrators, various system and anti-virus vendors. They provide the raw material for the rule engines of IDS, IPS and firewall systems.

The identity of the Honeypot is crucial and we can conclude that the learning curve (from the attacker) is directly proportional to the level of stealth for the Honeypot[4]. Research Honeypots are usually deployed at Universities and by the R&D departments of various organizations as High-Interaction Honeypots.

**Honeynet**

A Honeynet is a special kind of high-interaction Honeypot. Honeynets extend the concept of a single Honeypot to a highly controlled network of Honeypots. A Honeynet is a specialized network architecture configured in away to achieve:

- **Data Control**: It deals with the containment of activity within the Honeynet.

- **Data Capture**: It involves the capturing, monitoring and logging of all threats and attacker activities within the Honeynet

- **Data Collection**: Captured data is securely forwarded to a centralized data collection point.

This architecture creates a highly controlled network, in which one can control and monitor all kinds of system and network activity. Honeypots are then placed within this network. A basic Honeynet comprises of Honeypots placed behind a transparent gateway – the Honeywall. Acting as a transparent gateway the Honeywall is undetectable by attackers and serves its purpose by logging all network activity going in or out of the Honeypots.

**Virtual Honeypot**

Virtualization is a technology that allows running multiple virtual machines on a single physical machine. Each virtual machine can be an independent Operating System installation., running concurrently with others.

This is achieved by sharing the machine's physical resources such as CPU, memory, storage and peripherals through specialized software across multiple environments. This reduces project hardware costs. A virtual Honeypot is a complete Honeypot running on a single computer in virtual environment. Virtualbox Server[11] was used as the virtualization solution for our project. Virtualbox was selected because: It is free, reliable, has large community support, and extensive documentation. It is flexible and has robust networking components. For the scope of our project, we used Virtualbox Server version 3.0.2 for linux.

## TOOLS

### On the Gateway

For our project implementation the aim was to use free or Open Source tools. The Honeynet Project is a non-profit, Open Source security research organization. This organization has actively published papers, developed and contributed Open Source security tools. For the scope of our project we used dionaea. Dionaea is meant to be a nepenthes successor, embedding python as scripting language, using libemu to detect shellcodes, supporting ipv6. Dionaea intention is to trap malware exploiting vulnerabilities exposed by services offerd to a network, the ultimate goal is gaining a copy of the malware. Dionaea offers the following services by default, SMB (main service offered), HTTP, FTP, TFTP, MSSQL, MYSQL and SIP. Some features and working of the dionaea are discussed below:

### Libemu

- Detect shellcode, measure the shellcode, and even execute the shellcode.
- Runs the shellcode in a libemu VM, and records API calls and arguments.
- Multi-stage shellcode: First stage retrieves a second shellcode from the attacker.
- Allow the shellcode to take certain actions: e.g. create a network connection.

**Payloads:** Once we have the payload, and the profile, dionaea has to guess the intention, and act upon it.

**Downloads:** Once dionaea gained the location of the file the attacker wants it to downloads from the shellcode, dionaea will try to download the file. The protocol to downloads files via tftp and ftp is implemented in python (ftp.py and tftp.py) as part of dionaea, downloading files via http is done in the curl module - which makes use of libcurl's awsome http capabilities.

**Submit:** Once dionaea got a copy of the worm attacking her, we may want to store the file locally for further analysis, or submit the file to some 3rd party for further analysis.

**Logging:** Dionaea can write information to a text file, but be aware, dionaea’s logging to text files is rather chatty, really chatty, and you do not want to look at the information, if you are not debugging the software or writing some new feature for it. dionaea uses some internal communication system which is called incidents. An incident has an origin, which is a string, a path, and properties, which can be integers, strings, or a pointer to a connection.

### On the Honeypot

We implemented a standard Ubuntu server as our Honeypot. This server had basic services running that included SSHD, FTPD and HTTPD etc. Open SSH was patched and custom compiled to add both a user name and a password logging capability. The passwords were logged to a secure hidden directory within the server. A shell and perl script was used to capture the logs and associate them with attacker IP’s. Based on a timestamp difference, this script could generate logs and reports on hourly basis.

## PROPOSED ARCHITECTURE

The Honeynet Project provides some general documentation on deploying virtual Honeypots. The document was a How-To for deploying virtual Honeypots using VMware. This served as a standard template for anyone wanting to deploy a virtual Honeynet/hpneypot.

## Problem Identification & Solution

During our literature review it was decided to use(2) as the standard template for our project's implementation. We are decided to use virtualbox as our virtual honeypot and dionaea host was a ubuntu machine. Our project is to implement a low interaction honeypot to collect Attack Data using Multi-level detection capabilities.

## Solutions towards the Problem

Multi-level based attack data collection:

Attack data collection and logging into system events and network events in the form of system data and network, PCAP data.

- Operating system's vulnerabilities based attack data collection.

  Operating system's vulnerabilities:

  - Services ( port)
  - Applications
- Network Intrusion Detection Based Attack Data Collection.
  - NIDS
  - Emerging Threat Signatures

In the current state of art- existing low interaction honeypots do not have this kind of multi level attack data detection e.g. operating system vulnerabilities based as well as Network Intrusion Detection System (NIDS). We have also incorporated the latest Emerging threat signatures using current state of art Network Intrusion Detection System (NIDS). We have compiled the NIDS incorporated with these emerging threats. In earlier (existing) low interaction honeypots, these kind of components was not involved to detect the latest emerging attacks. Attack data correlation based on the alerts generated by NIDS was being performed at the network level. If there are attack evidences collected on system level, then we can correlate the events generated on network level to know what kind of that attack was.

## Design Details and Discussions

As we proposed we implement a honeypot to collect attack data using multi-level detection capabilities. During the research implementations, there will be implementation of attack data collection and analysis based on Linux Based Honeypot Technologies as well as investigations of collected attack data based on open source Intrusion Detection. Raw Network PCAP data will be investigated to gather the statistical information about the attacks. Linux Based Platform will be used for the implementation and for design and for development database we will implement Sqlite database. Collected attack data automatically will be seed as input to statistical analyzer which will analyse the network PCAP data. To monitor and follow the trends on malware infections it has deployed various honeypots in unused IP space. The various attacks were performed from a separate system in the same network.

The system had Windows XP. Nmap was launched and a scan performed. This exposed the various clients along with their IP address, operating systems running on them and opened ports. Each record corresponds to a packet that is received. In case of an attack, the tools used by the attackers usually scan down the whole network. It finds the operating system and active services running on the machines and then tries to break in or access files. All these activities require a lot of packets to be directed towards the victim system.

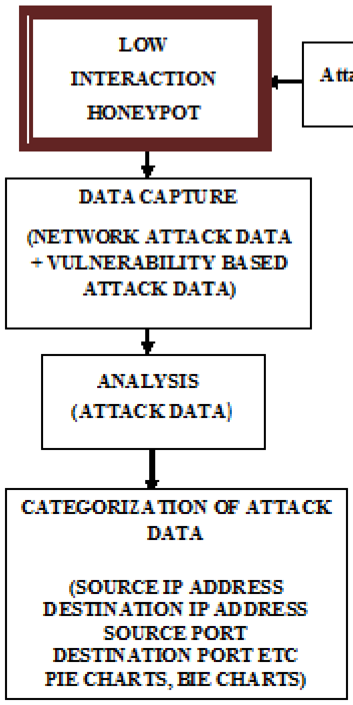

LOW INTERACTION HONEYPOT
Attacker
DATA CAPTURE
(NETWORK ATTACK DATA + VULNERABILITY BASED ATTACK DATA)
ANALYSIS (ATTACK DATA)
CATEGORIZATION OF ATTACK DATA
(SOURCE IP ADDRESS DESTINATION IP ADDRESS SOURCE PORT DESTINATION PORT ETC PIE CHARTS, BIE CHARTS)

**Figure 1: Various Modules of the Proposed Design**

Since honeypots are believed to be capturing mostly the attack activity almost all the source IP are suspicious. If a large number of records corresponding to same IP address and port are present in a certain time window then they are surely attackers. Figure1 shows the various modules and techniques used of the proposed design and here is the description of all these modules:

### Capture of Traffic

The network was scanned using Nmap which utilizes TCP/IP handshakes at the network level to administer connections and enumerate network information such as the operating system type and version, opened or blocked TCP/IP ports, in addition to active services. Dionaea captures all the packets passing through the network interface of the dionaea host and saves it in a form of ids_logs.

### Analysis Module

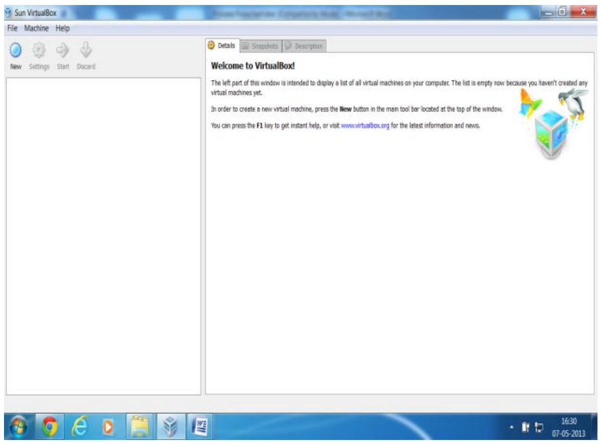
The logs captured by the dionaea is processed by the analysis module. There is a shell script which generated a report on hourly basis when system connected to the LAN. The records are inserted in a Mysql database. The output of the analysis module includes files for generating pie charts.

### Categorization of Data

The file generated by the analysis module is used as input to the PERL script to generate the categorized and graphical representation of the attack data. The pictorial representation is much easier to comprehend and have been used since a very long time by other analyzers as well.

## RESULTS AND DISCUSSIONS

Here we discuss the working and few experimental results to signify the proper working of our developed and implemented low interaction virtual honeypot. The virtual Honeynet was online for a period of approximately 30 days. Following sections describe in detail the resulting statistics established:



**Welcome to VirtualBox!**

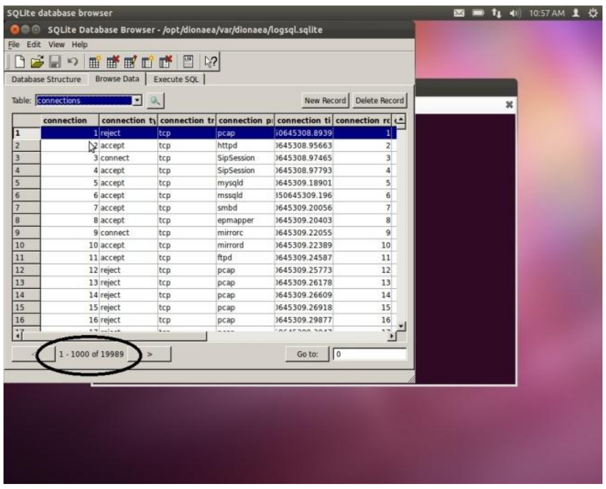
The left part of this window is intended to display a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet.

In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window.

You can press the **F1** key to get instant help, or visit www.virtualbox.org for the latest information and news.

**Figure 2: Screens Hot of Virtual Box**

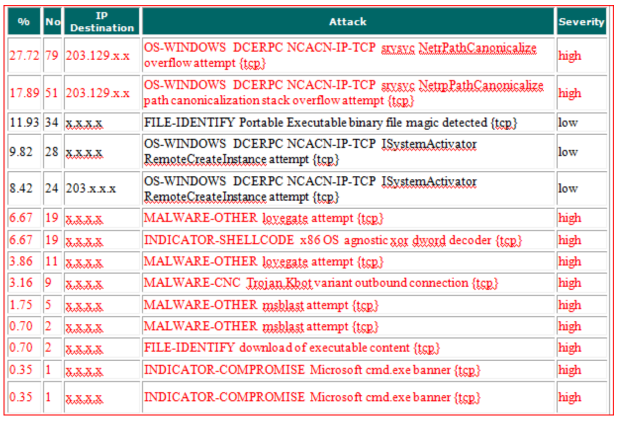
Figure 2 shows thr working snapshot of the system with Windows operating system on base machine, Linux operating system on virtual machine. This is established using virtualized environment with the help of Virtual Box tool; we can create multiple OS on a single machine which reduce the cost of hardware requirement.



| | connection | connection ty | connection tr | connection p | connection ti | connection rc |
|---|---|---|---|---|---|---|
| 1 | 1 | reject | tcp | pcap | 50645308.8939 | 1 |
| 2 | 2 | accept | tcp | httpd | )645308.95663 | 2 |
| 3 | 3 | connect | tcp | SipSession | )645308.97465 | 3 |
| 4 | 4 | accept | tcp | SipSession | )645308.97793 | 4 |
| 5 | 5 | accept | tcp | mysqld | )645309.18901 | 5 |
| 6 | 6 | accept | tcp | mssqld | 350645309.196 | 6 |
| 7 | 7 | accept | tcp | smbd | )645309.20056 | 7 |
| 8 | 8 | accept | tcp | epmapper | )645309.20403 | 8 |
| 9 | 9 | connect | tcp | mirrorc | )645309.22055 | 9 |
| 10 | 10 | accept | tcp | mirrord | )645309.22389 | 10 |
| 11 | 11 | accept | tcp | ftpd | )645309.24587 | 11 |
| 12 | 12 | reject | tcp | pcap | )645309.25773 | 12 |
| 13 | 13 | reject | tcp | pcap | )645309.26178 | 13 |
| 14 | 14 | reject | tcp | pcap | )645309.26609 | 14 |
| 15 | 15 | reject | tcp | pcap | )645309.26918 | 15 |
| 16 | 16 | reject | tcp | pcap | )645309.29877 | 16 |

**Figure 3: Screen Hot of Collected Data into Sqlite Database**

Figure 3 shows the logged data into the sqlite database of the virtual honeypot system. If any interactions occurs on ports and vulnerability services, i.e. if any nmap scan occurs it will be logged into Sqlite database. It logged all the details like connection type, connection protocol, local host and ports, remote hosts and ports etc. There is a GUI interface where we can see all the logged data into the database. Dionaea honeypot generate the logs which are normally network based attack data collection and ids logs.

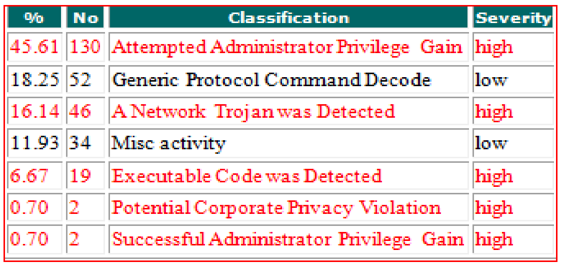


| % | No | IP Destination | Attack | Severity |
|---|---|---|---|---|
| 27.72 | 79 | 203.129.x.x | OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrPathCanonicalize overflow attempt {tcp} | high |
| 17.89 | 51 | 203.129.x.x | OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrpPathCanonicalize path canonicalization stack overflow attempt {tcp} | high |
| 11.93 | 34 | x.x.x.x | FILE-IDENTIFY Portable Executable binary file magic detected {tcp} | low |
| 9.82 | 28 | x.x.x.x | OS-WINDOWS DCERPC NCACN-IP-TCP ISystemActivator RemoteCreateInstance attempt {tcp} | low |
| 8.42 | 24 | 203.x.x.x | OS-WINDOWS DCERPC NCACN-IP-TCP ISystemActivator RemoteCreateInstance attempt {tcp} | low |
| 6.67 | 19 | x.x.x.x | MALWARE-OTHER lovegate attempt {tcp} | high |
| 6.67 | 19 | x.x.x.x | INDICATOR-SHELLCODE x86 OS agnostic xor dword decoder {tcp} | high |
| 3.86 | 11 | x.x.x.x | MALWARE-OTHER lovegate attempt {tcp} | high |
| 3.16 | 9 | x.x.x.x | MALWARE-CNC Trojan.Kbot variant outbound connection {tcp} | high |
| 1.75 | 5 | x.x.x.x | MALWARE-OTHER msblast attempt {tcp} | high |
| 0.70 | 2 | x.x.x.x | MALWARE-OTHER msblast attempt {tcp} | high |
| 0.70 | 2 | x.x.x.x | FILE-IDENTIFY download of executable content {tcp} | high |
| 0.35 | 1 | x.x.x.x | INDICATOR-COMPROMISE Microsoft cmd.exe banner {tcp} | high |
| 0.35 | 1 | x.x.x.x | INDICATOR-COMPROMISE Microsoft cmd.exe banner {tcp} | high |

**Figure 4: Distribution of Attack Methods**

Figure 4 depicts a table which is generated by our designed honeypot. There is a perl script written which takes logs generated by the dionaea as input and generate a report on hourly basis when connected to the LAN. This report shows the categorised and graphical representation of the attack data. This figure shows the attack and IP source of the attack and shows its severity also.

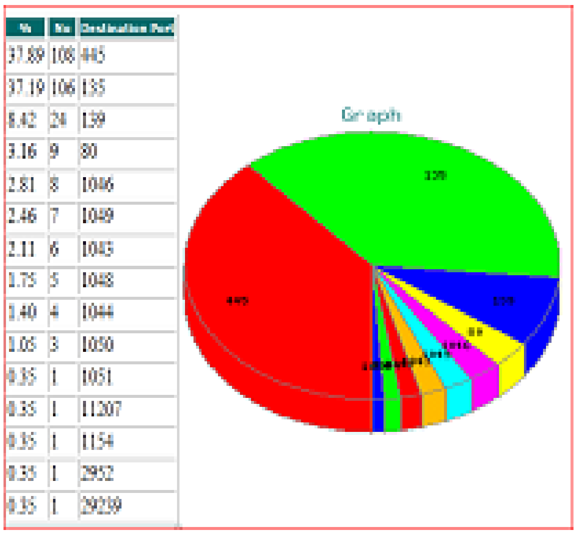
- **RED:** Dangerous connection (potentially bad, further investigation needed)
- **ORANGE:** Warning connection (Strange, may need further investigation)
- **BLACK:** Not dangerous alert (only low and unknown alert)



| % | No | Classification | Severity |
|---|---|---|---|
| 45.61 | 130 | Attempted Administrator Privilege Gain | high |
| 18.25 | 52 | Generic Protocol Command Decode | low |
| 16.14 | 46 | A Network Trojan was Detected | high |
| 11.93 | 34 | Misc activity | low |
| 6.67 | 19 | Executable Code was Detected | high |
| 0.70 | 2 | Potential Corporate Privacy Violation | high |
| 0.70 | 2 | Successful Administrator Privilege Gain | high |

**Figure 5: Distribution of Classification Method**

Figure 5 shows a table which distribute the attacks and represent that attacks according to their class and type.



| % | No | Destination Port |
|---|---|---|
| 37.89 | 108 | 445 |
| 37.19 | 106 | 135 |
| 8.42 | 24 | 139 |
| 3.16 | 9 | 80 |
| 2.81 | 8 | 1046 |
| 2.46 | 7 | 1049 |
| 2.11 | 6 | 1043 |
| 1.75 | 5 | 1048 |
| 1.40 | 4 | 1044 |
| 1.05 | 3 | 1050 |
| 0.35 | 1 | 1051 |
| 0.35 | 1 | 11207 |
| 0.35 | 1 | 1154 |
| 0.35 | 1 | 2952 |
| 0.35 | 1 | 29239 |

**Figure 6: Distribution of Event by Destination Port**

Figure 6 shows the distribution of events according to their destination port number. It shows a graph also which also shows the same distribution of events by their destination port number but in graphical manner. For quick view graph is the best way and tables are used for their all details. There are also some other tables and graphs generated like distribution of events by hours and protocols and according to the methods of the attacks generated. By adding the features in perl script we can create any number of tables, charts and graphs according to the requirement of the project you are using.

## SUMMARY

### Project Summary

The hardware and software used for this project has been summarized in Table 1. It can be inferred that standard hardware can easily be used to setup a virtual Honeypot. Large amounts of memory are always a preference for virtualized environments and can be used as a performance benchmark. The availability of free and Open Source tools and technologies such as Linux, Virtualbox, Snort, and those provided by the Honeynet Project [12] have considerably eased the process of setting up such projects. However, such tools demand a high degree of skill and customization, along with a thorough understanding of the system. Since there is no out-of-the-box Honeypot solution available these free and Open Source tools serve as very flexible and robust toolset for security engineers. The security design, policy and architecture may vary from organization to organization, based on their implementation.

**Table 1: Summary of the Project**

| **Hardware/Software Specifications** | | |
|---|---|---|
| **Feature** | **Product** | **Specs** |
| Host Operating system | Window 7 | **HW Vendor:** SONY Processor: 2.40 GHz Processor<br>**RAM :** 4GB RAM<br>**Storage:** 180 GB<br>**NIC :** 1 GB Ethernet controller |
| Guest Operating System | Linux Ubuntu | Single processor Virtual Machine<br>**RAM** 256MB<br>**NIC** 100Mbps host-only vmnet |
| Virtualization Software | VirtualBox | Virtualbox3.0.2 for linux |
| Architecture | Low Interaction Honeypot | Low interaction honeypot |
| Packet Capturing | TCPDUMP | Tcpdump tool/ Snort |
| Active Scanning | Nmap | Active scanning |

We successfully setup and maintained this project. A great deal was learned from this experience and certain areas for improvement have been identified. We have achieved a significant level of familiarity with all the tools utilized for the project and have identified some areas for further tool development and enhancement.

## CONCLUSIONS AND FUTURE WORK

During the course of our research work we implement a low interaction virtual honeypot which incorporate multi level detection capabilities and logged attack data in the sqlite database. The main motive is to generate a graphical report of the logged data which represent or categorize the attack data in the form of graphs and tables which are easy to understand and used multi level detected attacked logged data like NIDS and OS vulnerabilities based attack data. We used free and open source tools for our project like virtualbox and dionaea is used as a low interaction honeypot. Emerging threat signatures are used for better categorization of attacks.

The experimental results showed that we successfully done our work and this system is very useful to detect malwares and other useful information about the attackers.

Our future work will incorporate running virtual Honeypots/honeynets using other available Open Source virtualization software such as VMware ESX, Xen and VirtualBox[11] and performing a comparative study of these tools in terms of running Honeynets efficiently. We believe that there is a great need for enhancing and automating the data capture mechanism. The current mechanism lacks correlation of network and host events.

We feel there is a dire need to develop better analysis tools to efficiently and effectively correlate attacks with attackers and suggest prevention and detection techniques.

## ACKNOWLEDGEMENTS

We would like to sincerely thank Er. Vanita Rana,

Asstt Professor, Department of Computer Science for her contribution and help in writing this paper.

## REFERENCES

1. Sun Bing "Study of Network Security Situation in Honeynet," Proceedings of International Conference on Modelling, Identification and Control, Wuhan, China, June 24-26, 2012.
2. A.N Singh, "A honeypot system for efficient capture and analysis of network attack traffic," International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011.
3. Yogender Kumar Jain, Surabhi Singh, "Honeypot based Secure Network System," International Journal on Computer Science and Engineering (IJCSE), Vol. 3, Feb 2, 2011.
4. Fahim H. Abbasi, "Experiences with a Generation III virtual Honeynet," Telecommunication Networks and Applications Conference (ATNAC), 2009 Australasian.
5. David Watson, "The Honeynet Project: Data Collection Tools, Infrastructure, Archives and Analysis," WOMBAT Workshop on Information Security Threats Data Collection and Sharing, 2008.
6. Chao-Hsi Yeh, "Design and Implementation of Honeypot Systems based on open-source Software," IEEE International Conference on Intelligence and Security Informatics, 2008.
7. Haifeng Wang, "Design of cooperative deployment in distributed Honeynet system," 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD), April 14-16 2010.
8. Know Your Enemy: GenII Honeynets, Easier to deploy, harder to detect, safer to maintain. Honeynet Project, http://www.honeynet.org , Last Modified: 12 May, 2005.
9. http:// www.honeyd.org/background.php.
10. Nmap. [Online]. Available: http://www.insecure.org/nmap/nmapfingerprinting-article.html.
11. VirtualBox. (2004). Sun VirtualBox® User Manual, Available: http://www.virtualbox.org/manual/UserManual.html.
12. The Honeynet Project,1999".