Distributed Honeynet System Using Gen III Virtual Honeynet

Sanjeev Kumar, Paramdeep Singh, Rakesh Sehgal, and J. S. Bhatia

***Abstract*—In this paper , with the reference of many problems in current traditional security resource applications, and based on the research on Honeypot Technology, the Honeypot Technology is used in network security defence, and a Honeypot based Distributed Honeynet System is presented. This paper presents a unique method for set up and establishment of Distributed Honeynet System at various Geographical locations using Gen III virtual Honeynet which is running Honey wall CDROM Roo. We are specifically using Linux based host in our current implementation which have single physical network Interface card(NIC) and a large number of virtual Honeypots as Guest OS on the single base operating system.. Three tier based Distributed Honeynet System is presented which is dynamically configurable in terms of IP, services and OS. Further automated botnet command extraction based analysis is presented. We are ensuring that our solution is completely automated.**

***Index Terms*—Computer security, malware, network security, honeyot, honeynet.**

I. Introduction

The growth of Information Technology is very revolutionary in terms of large applications and services in heterogeneous and high speed networks. The business of enterprises based on the top of these high speed and heterogeneous networks. Networks have been changed from low speed to gigabit stream networks. In today's highly networked and extremely heterogeneous network computing environment, the increasing sophisticated exploitation of security flaws has become a significant problem for private users, business, and even government [1]. Generally attackers pretend to gain control as many system as possible, The attacker may injects a malware program of different type such as trojan, back-doors or spyware programs through a security flaw or user intention into the attacked system.

By learning the techniques used by the black hats crackers we can secure these kinds of high speed infrastructures, services and applications running on them. With the help of honeypot we can learn the techniques used by the attackers through which they are able to gain the legitimate access to system resource along with techniques for analysing the tools they are used to obtain this access [2]. We can monitor and log the attacker's additively by providing them the vulnerable environment, further we can study the motivation, tools and techniques used by the attackers to launch the attacks.

Manuscript received May 15, 2012; revised June 20, 2012.

Sanjeev Kumar is with Center for Development of Advanced Computing, Mohali, India (e-mail: ror.sanjeev@gmail.com).

Paramdeep Singh and Rakesh Sehgal are with Cyber Security Technologies Division, Mohali, India.

If we have a look into most of the network security devices like firewalls, IDS etc, they are usually passive in nature based on known signatures and associated rules in their database [3], [4], [5]. Only with the help of these associated rules in their knowledge base, they are able to detect anomalies occurred. Then what happens if any kind of activity that does not match with these associated rules that go undetected. That is the place where honeypots are required. A Honeypot is a system that is used to detect and analyse the attack performed by the attacker The Honeypot has no intervention with the production traffic, therefore anything which comes on the honeypots is most likely the malicious intent [6]. As compare to any other available security tools, Honeynet are capable of logging far more information. They provide the environment, flexibility to the attackers so that they can attack on the information system and every aspect of them is logged and can be analysed. To reduce considerably the hardware cost, virtualization technology like VirtualBox [7] an Open Source virtualization product, provide the flexible environment to set-up the network with single physical machine. We can run multiple guest OS on a single machine running with Linux operating system.

The remaining paper is described as follows: section II, defines and explains the technology that has been employed and discusses the evolution of Honeynet in brief. Section III deliberates the problem statement and discusses our proposed approach and details of implementation. Section IV measures the effectiveness of Distributed Honeynet System using virtual Honeynet by investigating data that has been collected and correlate it with attacks and suspicious flows. Section V describes the ideas for future road and technology.

II. Background and Motivation

A. Honeynet

A Honeynet is a special kind of high-interaction Honeypot. Honeynets expand the concept of a single Honeypot to a highly controlled network of Honeypots. A Honeynet is a specialized network architecture configured in a way to achieve: Data Control, Data Capture, Data Collection and Data Analysis [8], [9].

Data control deals with the containment of activity within the Honeynet.

Data Capture deals with the monitoring and logging of all the activities within the Honeynet.

Data Analysis: It deals with the analysis of collected data on Honeynet.

Data Collection exists with the organisation which have

large number of Honeynet in Distributed environment which are geographically deployed at various locations have to collect all of the captured data and store it in a central location.

B. Virtual Honeynet

With the virtualization technology we can run multiple machines on a single physical machine like each virtual machine can be an independent Operating System [9]. This is achieved by the sharing the machine's physical resources such as memory, CPU, storage and peripherals through specialised software across multiple environments. With the help of virtualisation, definitely we can reduce the hardware cost as we are running just like different real operations system on single base machine. VirtualBox [7] have been used in our project.

III. Proposed Architecture

In this section we are discuss the architecture proposed and respective tools and techniques used in our Distributed Honey net Architecture. In this paper, we are proposing a Distributed Honeynet System architecture which is completely automated system. The collected set of binaries are automatically stored in central database which are further automatically fetched and submitted to the analysis server for further analysis of the captured malwares. As we have discussed earlier, we are using the hybrid honepots in vrtual Honeynet. Hybrid Honeypots are a combination of low interaction honeypot sensors such as nepenthes, and high interaction Honeypots sensors like Windows XP, Linux etc. Low interaction Honeypots provide the emulated kind of environment to the attackers with a set of vulnerabilities. In contrast, High Interaction Honeypots provide the real OS environment to attackers. Fig. 1 show the Distributed Honeynet Architecture. As presented in the architecture, we discuss the components of Distributed Honeynet System keeping in mind with various tools and techniques used; the following architecture includes the major components such as: Distributed Honeynet Client Node, Central Database server and Analysis server. We have collected the malware by distributed deployment of malware collection framework which is DHS client node, the collected malware are further sent to the central database server, which further automatically fetched and analyse by the Analysis server.

A. Distributed Honeynet Client Node

For the implementation of Distributed Honeynet client node, we have chosen to use Virtual box [7] for the virtualization which is an Open Source Virtualization product. By using virtualization, it will reduce the hardware cost as compare to real physical system as well as will improve the deployment and maintenance.

As shown in the Fig., for the malware collection using client node, we have used the mixture of low interaction such as nepenthes and high interaction honeypots such as Windows XP, Linux. For the data control, limiting the inbound and outbound traffic, we are using Honeywall Roo 1.3[6] as gateway for the high interaction honeypots Lan. All the traffic generated from the high interaction honeypots Lan has to pass through the Honeywall gateway which is also running in a Virtual Machine All the high interaction honeypots are automatically rebuild from a clean snapshot. We take the differences with a clean image and send all the executables found to our central server for further analysis. Following flowchart show the flow of our rebuilding images. All the files Added, Modified or Removed are compressed and then send to the central server for future analysis.

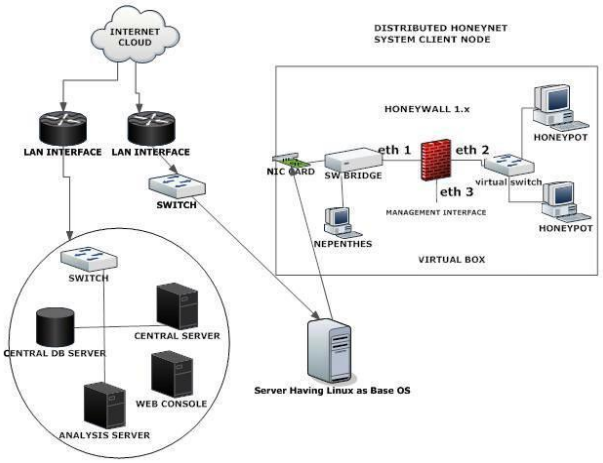

Fig. 1. Distributed honeynet architecture.

B. Central Database Server

All the executables which are potentially malwares are stored in the database. We are collecting malware binaries which are unique according to their MD5 value and stored as binary fields in central database. For the database implementations, we are using the MySql. Fig. 2 shows ER diagram of the database.

C. Analysis Server:

Now we come across our analysis server on which we are running our code for further analysis of malware binaries. We are using deep packet based (DPI) algorithm for analysis of PCAP dump file [10].

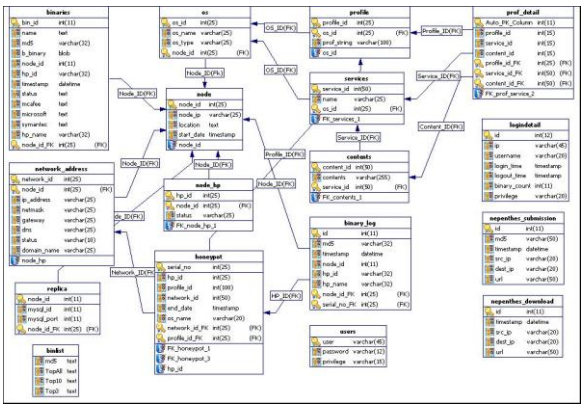

Fig. 2. E-R diagram.

We are retrieving the PCAP dump files for each geographical node and give single/multiple PCAP dump files as an input to our code which basically converts them into composite payload. That composite payload is actual data stream exchanged in communications from which we are extracting command signatures used in botnet communication. At present we are focusing our analysis to Botnet commands [11]-[16] extraction in bot and Botnet C and C server communication. We are able to extract various

live botnet commands exchanged during communications. We can use do more analysis of corresponding malware samples but we are restricting ourselves for only botnet command extraction. Our complete analysis system is automated.

Below is the generic algorithmic steps used in our analysis algorithm:

1) Give single/multiple PCAP dump files as an input to our automated PCAP parser.
2) If PCAP_new=PCAP_old {Retrieve the old generalised payload from database and submit it to payload parser for extraction of live commands exchanged in bot communications.} Else {Parse the PCAP_new and generate the actual data payload, Submit generated payload to payload parser for extraction of live commands exchanged in bot communication.}
3) Repeat step 3 for each PCAP dump file.
4) Generate report of botnet tracking and command exchanged.
5) Give user access to download payload files and botnet report.

```
#part of payload generated code
  while(fgets(dline,128,fp)!=NULL)
      {
            int len = strlen(dline);
            if(dline[len-1] == '\n')
 dline[len-1] = 0;
            bzero(buff,strlen(buff));
            sprintf(buff,"find '%s/pcap/%s' | egrep log |
sort >temp/logfile.txt",path,dline);
            system(buff);
            FILE *fp1=fopen("temp/logfile.txt","r");
            while(fgets(fline,128,fp1)!=NULL)
            {
                  int len = strlen(fline);
                  if(fline[len-1] == '\n')
                        fline[len-1] = 0;
                  bzero(buff,strlen(buff));
        printf("path: %s\n",path);
                  sprintf(buff,"tcpflow -r '%s'
-c >> %s/payload/payload",fline,path);
        printf("path: %s\n",path);
                  system(buff);
                  bzero(fline,strlen(fline));
            }
            bzero(dline,strlen(dline));
            fclose(fp1);
      }
```

IV. Results and Experiments

We built a test bed of Distributed Honeynet System using various high interaction honeypots and nepenthes sensors. We used high interaction honeypots like Windows 2000, Windows XP, unpatched with default configurations.We deployed the Distributed Honeynet client node at 8 Geographical locations. During the operation, we have detected more than 14000 samples (about 1300, unique samples, 50GB of PCAP data collected. With this distributed deployment, we have observed if any unique IP address were seen on multiple Distributed Honeynet Nodes, not all 8 nodes were live for entire data collections but we are able to detect 19 unique IPs seen by 5 distributed nodes, 24 number of unique IPs seen by 4 distributed nodes, 29 unique IPs seen by 3 nodes and 80 unique IPs seen by 2 distributed nodes. Top source was Egypt bases with 11200 numbers of flows but others like India, China were also there among top of the list. Fig. 3 depicts the country-wise distribution of Unique IP Distribution of Argus Flow [17].

We have basically concentrated on the detection of botnet. We are able to detect live botnet communication. The majority of the IRC botnets are using commands like PING, PONG, JOIN, PRIVMSG and most of the attack specific commands found were DDOS, ASC, and VSCAN. In most of the scan activity we found were ICMP scan or random port scanning. We are also able to detect continuous SYN flooding to random foreign IPs. Below is one of the live communication captured and which is declared IRC bot by most of the Anti-virus products.

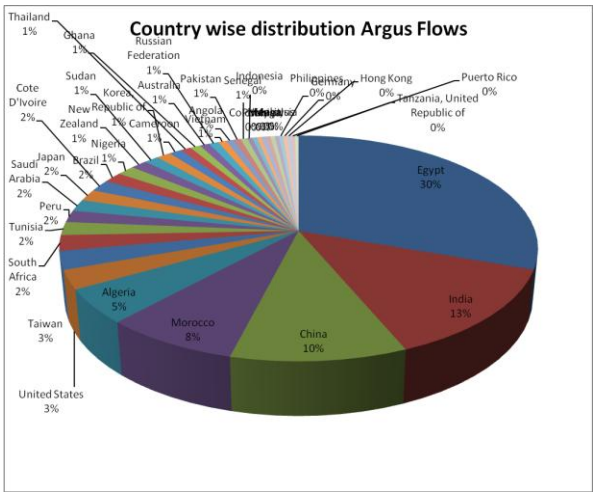

Fig. 3. Country-wise distribution of IP flow.

Symantec: W32.IRCBot, Microsoft: Backdoor: Win32/Poebot

```
PASS 146751dhzx
:ftpelite.mine.nu
NICK kcrbhf8wlzo
USER XPUSA6059014236 0 0 :o4dfmj2ctyc
:ftpelite.mine.nu
PING: AE645AF3
PONG AE645AF3
:ftpelite.mine.nu 332 kcrbhf8wlzo #100+ :| .vscan netapi 50 5
9999 216.x.x.x | .sbk windows-krb.exe | .sbk crscs.exe | .sbk
msdrive32.exe | .sbk woot.exe | .sbk dn.exe | .sbk
Zsnkstm.exe | .sbk cndrive32.exe |
```

PRIVMSG #100+:.4[SC]: Random Port Scan started on 216.x.x.x:445 with a delay of 5 seconds for 9999 minutes using 50 threads.

```
PASS 146751dhzx
:ftpelite.mine.nu
NICK kcrbhf8wlzo
USER XPUSA6059014236 0 0 :o4dfmj2ctyc
:ftpelite.mine.nu
```

PRIVMSG #100+:.4[SC]: Random Port Scan started on 216.x.x.x:445 with a delay of 5 seconds for 9999 minutes using 50 threads.

```
PRIVMSG #100+: BotKill Started: windows-krb.exe
PRIVMSG #100+: BotKill Started: crscs.exe
```

PRIVMSG #100+: BotKill Started: msdrive32.exe
PRIVMSG #100+: BotKill Started: woot.exe
PRIVMSG #100+: BotKill Started: dn.exe
PRIVMSG #100+: BotKill Started: Zsnkstm.exe

And below shows the network activities performed by one of the bot sample:

Network Activities:

NICK USA|XP|SP2|072498
USER SP2-095 * 0: -3B35342B0F
:. 332 USA|XP|SP2|072498 #naga4:|.ddosstop -s|.stop -s|.sftp 2689 123 123 gff6.exe -s|.asc svrsvc_SP2 100 5 9999 1 -b -e -r -s|.asc svrsvc_XXX 100 5 9999 1 -b -e -r -s|.join #sd1 –s
Activity Observed: ICMP scan

List of IPS Scanned: 124.232.X.X, 89.193.X.X, 202.77.X.X, 14.244.X.X, 202.15.X.X

Fig. 4 shows the network activities captured in wireshark tool [10] and as shown it is performing random and sequential port scanning with private messages (PRIVMSG).

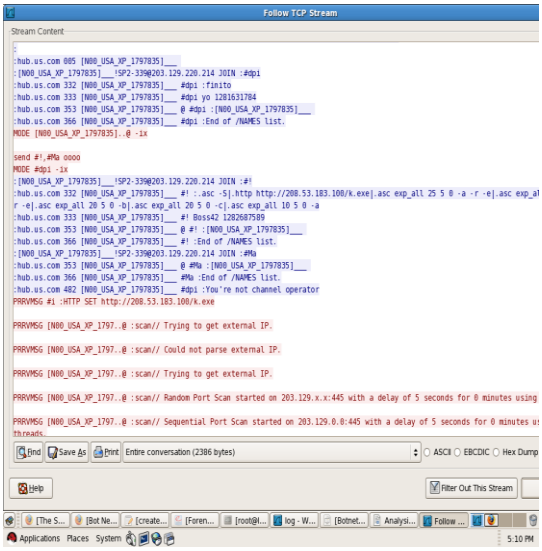

Fig. 4. Network communication.

We are also able to detect some samples which are undetected by most of anti-virus products, below are the actual communication seen in one of the sample which is undetected by the most of anti-virus products:

- Binary Name : B101.exe
  - BOT hunter result: not valid win32 error
  - BOT detection engine: Not BOT
  - Symantec: No result
  - Type : IRC
  - Activity :.asc-S|.http http://208.x.x.x/k.exe|.asc exp_all 25 5 0 -a -r -e|.asc exp_all 25 5 0 -b -r -e|.asc exp_all 20 5 0 -b|.asc exp_all 20 5 0 -c|.asc exp_all 10 5 0 -a

Project Summary

The summarization of various software and hardware used in our project is depicted by the following table. In this project we are ensuring that most of the tools are free and open source. For Virtualization technology, we are using Virtual Box [7]. We are running different Virtual machines on single Red hat linux based Operating System. We are using minimum memory of 4GB but large amount of memory is preferred to run Virtual Machines in Virtualized environment.

TABLE I: PROJECT SUMMARY.

Project Summary		
Feature	**Product**	**Specs**
Host Operating system	Red Hat Enterprise Linux 5	**Hw Vendor: HP Server Proliant ML 350 Processor: 2.33 GHz Processor RAM :**4GB RAM **Storage:** 2x146 GB **NIC:** 1 GB Etherenet controller (public IP)
Guest Operating System 1	Linux,honeywall Roo 1.3	Single Processor Virtual Machine **RAM** 512 MB **NIC 1**: 100Mbps Bridged interface vmnet0 **NIC 2**: 100 Mbps host-only interface vmnet1 **NIC 3:** 100Mbps Bridged interface Vmnet2(Public IP)
Guest Operating System 2	Red Hat Enterprise Linux 5	Single processor Virtual Machine **RAM** 256MB **NIC** 100Mbps host-only vmnet(public IP)
Guest Operating System 3	Microsoft XP SP2	Single processor Virtual Machine **RAM** 256MB **NIC** 100Mbps host-only vmnet(public IP)
Guest Operating System 4	Nepenthes	Single processor Virtual Machine **RAM** 256MB **NIC** 100Mbps host-only vmnet(public IP)
Virtualization Software	VirtualBox	Virtualbox3.0.2 for linux
Architecture	Gen III	Gen III Implemented as a virtual Honeynet
Honeywall	Roo	Roo 1.3
IDS	Snort	Snort 2.8.3
IPS	Snort_inline	Snort_inline 2.8.3
Data Capture Tool	Sebek	Sebek 3.2.0

V. CONCLUSION AND FUTURE WORK

We have proposed a Distributed Honeynet Architecture which is having capability of dynamically reconfigurable in terms of IP, OS and services and completely automated including analysis of Botnet communications. We believe our solutions, if widely deployed, could significantly ease the sharing of collected data. We are having large amount of malicious PCAP data which is further useful for research perspective and can serve as a excellent environment for development of an automated IDS signatures. Our solution is completely automated but lack of automated correlation of attacker source IP address to Sebek Keystrokes remains a major problem [18], [19]. Our database schema is presently only for centralized botnets; no support for P2P botnets and encrypted botnets. We plan to add some basic support for these kinds of botnets also.

ACKNOWLEDGMENT

We would like to thank Malware collection team of Cyber Security Technology Division at CDAC, Mohali to provide the useful help in collecting the malwares to make them available for further analysis. We also very thankful to Executive Director of CDAC, Mohali to provide us full support.

REFERENCES

[1] Security threat report: Sophos Group. (2010). [Online]. Available: http://www.sopos.com/security/topic/secuirtyreport-2010.html
[2] L. Spitzner, *Honeypots: Tracking Hackers*, US: Addison Wesley, pp 1-430, 2002.
[3] SNORT - The de facto standard on Intrusion Detection and Prevention. (2006). [Online]. Available: http:// *www.Snort.org*.
[4] Snort user manual 2.8.3. [Online]. Available: http://www.Snort.org
[5] W. Stallings, *Cryptography and Network Security Principles and Practices*, Third Edition, Prentice Hall, 2003.
[6] The Honeynet Project. Know Your Enemy: Honeywall CDROM Roo. (2008). [Online]. Available: http://old.honeynet.org/papers/cdrom/Roo/index.html.
[7] SUN Microsystems. VirtualBox. [Online]. Available: http://virtualbox.org/
[8] V. Padmanabban and L. Subramanian, "Determining the geographic location of Internet hosts," *SIGMETRICS/Performance*, pp. 324– 325, 2001.
[9] F. Abbasi and R. Harris, "Experiences with a Generation III virtual Honeynet," *Telecommunication Network and Applications Conference (ATNAC)*, 2009.
[10] Wireshark. [Online]. Available: http://www.wireshark.org
[11] M. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach tounderstanding the botnet phenomenon," in *Proceedings of the 6th ACM SIGCOMM*
[12] C. Stoll, *The Cuckoo's Egg: Tracking a Spy through the Maze of Computer Espionage*, Pocket Books, New York, 1990.
[13] Virtual Box. Sun Virtual Box® User Manual. [Online]. Available: http://www.virtualbox.org/manual/UserManual.html
[14] P. Barford and V. Yegneswaran, "An Inside Look at Botnets," *Advances in Information Security*, vol. 27, pp. 171–191. Springer, US , 2007
[15] B. Edward and C. Viecco, "Towards a Third Generation Data Capture Architecture for Honeynets," in *Proceedings of the 2005 IEEE, Workshop on Information Assurance and Security, United States Military Academy, West Point, NY*, vol. 1, no. 1, pp. 21-28, 2005.
[16] *The Honeynet Project. Know Your Enemy: Tracking Botnets*, Internet March 2005.
[17] *Argus project*, 2004.
[18] Know Your Enemy: Sebek, A kernel based data Capture tool, The Honeynet Project. (2003). [Online]. Available: http://www.honeynet.org
[19] P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. Freiling, "The Nepenthes Platform: An Efficient Approach to Collect Malware," *Zamboni, D., Krügel, C. (eds.) RAID 2006. LNCS*, vol. 4219, pp. 165–184. Springer, Heidelberg, 2006.

Sanjeev Kumar received his B.Tech from Kurukshetra University Haryana, INDIA. He is working as a staff scientist at CDAC, Mohali, India. He is CCNA and CCNP certified and an active member of Indian National Grid known as GARUDA. His technical expertise are in networking, network security, network forensics, linux.

Paramdeep Singh received his MCA from PTU Punjab INDIA. Currently he is working in Cyber Security Technologies Division at CDAC Mohali, INDIA. He has 6 year extensive work experience on Honeynets and Honeypots.He Is RHCE certified.His Technical Expertise is in field of Information Security and Network Security

Rakesh Sehgal received his B.E in Electronics from Nagpur University, 1988 and M. Tech. in Computer Science from DAU, Indore. He is currently Principal Design Engineer and Head of Cyber Security Technology Division at CDAC- Mohali. He has vast research experience in Network Security, Honeynets and Honeypots.