Honeysand: An Open Source Tools Based Sandbox Environment for Bot Analysis and Botnet Tracking

Saurabh Chamotra
C-DAC,Mohali

Rakesh Kumar Sehgal
C-DAC,Mohali

Raj Kamal
DAV, Indore

ABSTRACT
Malware analysis is a process of determining the intent and modus operandi of a given malware sample. It is the first step in process of developing any preventive or defensive measure against a malware attack. The work presented in this paper is focused on the dynamic malware analysis. Dynamic malware analysis is one of the malware analysis techniques, in which the malware sample is executed in a controlled environment called sandbox and the effects of the execution at different levels of system abstractions (I.e. operating system, network, or kernel) are captured, stored and processed. In this paper we are presenting the design details of a malware execution environment named as Honeysand. The presented solution is specifically designed for catering the needs of performing dynamic analysis for a class of malwares known as bot. Bot is a class of mwalre that have the ability to coordinate among themselves and create a network of infected systems which is under the control of a single machine called command & control server [18] .Based upon the proposed system design we have developed a prototype system using the honeypot technology as a base with some other open source tools configured over it and used this prototype to demonstrate the effectiveness of the proposed solution.

1. INTRODUCTION
Malware are among one of the major threats to computer Industry. Due to their ubiquitous nature and degree of impact they have on the IT industry, they have gained lot of attention among both the researcher's and defender's community. As the foremost task in the process of the mitigation of a threat is its characterization. Hence in case of threat posed by malwares it is equally necessary for a defender to perform malware characterization using appropriate analysis tools and techniques. The outcome of a malware analysis process not only helps the defenders to characterize the malware and find out its intent but also is helpful in the development of the preventive and defensive measure against the malware. That is the reason why most of the signature based malware detection systems are dependent upon these malware analysis techniques to keep their signature database updated against the latest malware threats. The malware analysis techniques can be broadly categorized in to two main categories, static and dynamic malware analyses. Each one has its own strengths and weaknesses. The static malware analysis is performed without executing the malware, analyzing the malware file with the help of tools such as the dissemblers, debuggers and generating the control and data flow graphs [9][20][21]. Where as in dynamic malware analysis the malware sample is actually been executed in a controlled environment popularly known as sandbox. During this execution the traces of malware execution are captured at different levels of system abstractions by the stealth capturing sensors and using this data different behavioral reports are generated. In this dynamic malware analysis the sandbox environment plays a very crucial role as it takes care of things like malware execution confinement, deception and stealth capturing of important behavioral logs. In the work presented in this paper we have suggested the design of a low cost sandbox created using honeypot technology and open source tools. The proposed sandbox is specifically designed to cater the needs of Bot malware analysis. The word "Bot" addresses to a class of the malware that have the special characteristic to coordinate among them and create a network of bot infected machines, such networks are known as botntes.A normal botnet may have few thousands to millions of infected bots. All the infected machines in a botnet are controlled by a single machine that is known as C&C server or command and control server [27].The analysis for the detection of such bot malware samples and tracking of the related botnet have some special requirements that are not fulfilled by most of the available sandbox solutions. In the solution proposed in this paper we have added certain features specific bot analysis requirements. The proposed system named as Honeysand has following features.

1. The system is developed using open sourceTools
2. Capability to mimic the internet space
3. Limited connectivity to real internet
4. Popular services emulation
5. Specifically designed to cater the needs of bot analysis and botnet tracking.

2. RELATED WORK
In the work presented in paper [19] author has proved that the static code analysis based malware classification techniques followed by most of antivirus solutions ranks low in term of completeness, conciseness and consistency. The researcher who support static analysis have proposed many novel techniques such as semantics aware analysis [24][25][26] to overcome the demerits of static analysis. Although these techniques solves some of the problems related to static analysis but as demonstrated by [8] that it is still possible to evade these techniques. Due this, Dynamic malware analysis has gained a lot of attention among the researcher community which has resulted in various online dynamic malware analysis frameworks [1][2][3][4]. Most of the literature we came across during our research was either focused on efficient data capturing [7][13][14] or on vitalization detection [6][15][11].Whereas our work is related to the bot malware analysis hence the main focus is the network part. As bots are dependent upon the network inputs sent by their bot masters in the form of bot command to trigger any action hence the network characteristics of a bot malware is of great importance. In the work presented in [16] it has been brought in to the notice that open internet connectivity to a sandbox could leads to sever situations such as:

1. Detection and blacklisting of the sandbox IP address by the C&C servers
2. Sandbox getting attacked through internet.To avoid such situations concepts like
3. limited internet connectivity

4. Mimicking the internet

were introduced. Mimicking of internet was a concept first introduced by the miwa in their papers[6].The work proposed in the current paper shares some similarity at conceptual level with the work presented in [16][6].The main objective of honeysand is to provide cost effective easily scalable solution for the bot malware analysis and botnet tracking. It is created using open source tools and further uses the honeypot technology as the core of the sandbox architecture. Also not much attention is been paid to the issues related to the virtualization detection as with improved support for the virtualization in modern processors and the increased usage of virtualization in the main stream; virtulization detection is now no more a confirmation of the fact that the malware is being executed in the sandbox. This has also been observed by Chen.et.al [15] and Lau [11] et al.In the experiments performed by them hardly 4% to 2.5 % of malware possess the virtualization detection capability.

3. STATIC VS. DYNAMIC MALWARE ANALYSIS

Although static malware analysis technique is much faster than that of dynamic malware analysis but still it has many drawbacks. As the code that is analysised during the static analysis is not always the one that actually gets executed hence the results of the static malware analysis are always doubtful. Also there is plethora of techniques such as

- Code packaging
- Code obfuscation
- Encryption

extensively followed by the code writers to armor the static analysis process [21][22][23]. These techniques transform an executable into a syntactically different but semantically same representation. This has led to a scenario where we have diverse range of malwares class addressing to malware samples having almost same functional characteristics. The reason behind is fact that as most of the mechanisms used to detect and nomenclate malwares are based upon the static malware analysis technique. The static analysis only captures the syntactic features of the malware which can be easily altered through code syntax change without actually changing the functionality of the malware [25][26].Hence every time a malware author dose some changes in the malware code to bypass the detection mechanism we have a different class of malware sample created. Table 1 show the results of an experiment where we have performed dynamic malware analysis on a set of 400 malware samples. It was observed that for a subset of malware sample which have shown a common behavior (IRC bot) different antivirus solutions have categorized them in to different malware classes.

Table 1

Antivirus	Classes	Dynamic analysis Results
McAfee	Trojan.Gen, Packed.Generic.300 W32.Spybot.Worm W32.Esbot.A W32.Pilleuz Trojan Horse Trojan.Gen W32.Spybot.Worm W32.Rahack.H W32.IRCBot W32.Linkbot.M	IRC bot
Symantec	Win32.IRCBot, W32/Nirbot.worm W32/Sdbot.worm BackDoor-EEF W32/Virut.gen.a Undetected	IRC bot
Trendmicro	BKDR_RBOT.GFP BKDR_NEPOE.CW WORM_IRCBOT.BZT PE_VIRUX.GEN-3 WORM_VANBOT.QM TROJ_LETHIC.SMA BKDR_RBOT.GFP BKDR_VANBOT.AHH	IRC bot

Even though the behavioral based dynamic malware analysis techniques are slower as compared to the syntactic based approaches (static malware analysis) but still they give more reliable and accurate results. The reason is even if the malware authors uses the malware armoring techniques such as code packers, code obfuscators etc [9][10] to hide the code semantics , the overall functionality of the malware code is not changed much.

4. DESIGN OF HONEYSAND

The prime objective for the development of the honeysand is the creation of a system for detection of the bot malwares and tracking of botnets. The dynamic malware analysis provides an effective platform to extract the information such as unpacked code [9][10] botnet command and control (C&C) server IP addresses and signatures of communication traffic between bot binary and C&C server [18].

While designing a sandbox environment there are certain design considerations mentioned in [17] which has far reaching consequences on the system efficiency, scalability, accuracy and throughput. Following are some of the critical design issues that were considered by us.

- choice of the execution privilege level of the analysis component
- choice of execution environment i.e emulated,virtulised
- Network simulation
- System recoverability

In the design proposed by us we have taken following decisions

Table 2

execution privilege level of analysis component	**User space**
Choice of execution environment	Virtualization
Network simulation	A hybrid system of mimicked internet and limited connectivity
System recoverability	Virtual machine snapshots

Virtualization has gained lot of popularity which has resulted in its increased usage in IT mainstream. Due to which the virtualization detection is now no more a confirmation of presence of sandbox environment. Keeping this point in view we have opted for virtualization for the creation of malware execution environment.

Also as our main objectives is to extract the network based parameters form the bot binary sample and used it for the tracking botnets hence more consideration is given to the emulation of network characteristics. In the design of the network environment we have opted for a hybrid kind of network environment system which provides us the benefits of both

1. Mimicked Internet

The complete internet space is emulated for the malware.

2. Filtered internet connectivity

The malware sample is provided the limited internet connectivity.

The design Specifications of the systems are:

4.1 Mimicked internet

as most of the bot samples first tries to connect to a random IP address and only if they find some response they proceed further. Hence for those types of bot samples the fake internet is created which has the following capabilities

- Emulating any given range of IP addresses
- Sending ICMP echo reply packets
- Performing three way TCP hand shake signal
- Emulating all the ports in open, close states.

4.2 Limited connectivity to internet

as some of the bot first uses the services like time server (123), DNS (53), HTTP (80) to validate the genuineness of the internet provided to them. only in case they are able to perform these tests successfully (i.e. resolve the IP address, access a legitimate web site, access the correct time from a time server) They perform further activity. Hence one has to provide limited connectivity to legitimate services and genuine internet connection requests. In Honeynsand this point has been well taken care in a way that limited internet connectivity is being provided which is monitored and is well under the control of the administrator. To keep the danger of direct internet connectivity low we have designed the system in a way that the only legitimate necessary internet connections are allowed.

4.3 Network Services Emulation

Certain services such as

1. Ftp
2. Telnet
3. SSH
4. SMTP
5. POP

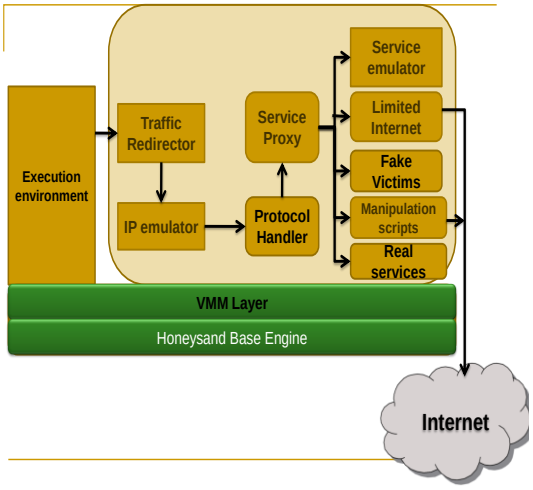
are been emulated to fetch the information such as login password Username and the commands that are executed by the malware after login in to the service. This information helps in knowing the motive of the malware and to create its behavioral profile. The emulated services are specially crafted scripts that provide the malware the fake login page and banner information

4.4 Manipulation scripts for server replies

During the dynamic malware analysis the malware behavior is observed for certain time period called observation window. In some cases there is a need of the certain manipulations to be done at network level to trigger the behavior of interest in the given observation window period. i.e sometime when an old malware sample is reanalyzed and it tries to resolve a set of the C&C domain names which are closed or not functional. In that case it will not receive any response that it was expecting and hence it ceases to execute and show the bot behavior in the observation window period. Hence there is need of a system that is able to manipulate the network replies sent to the malware in a way to get the activity of interest triggered in the given window period. In the current paper we have presented the instance of such kind of problem and a solution to it.

5. SYSTEM IMPLEMENTATION

With the above mentioned design objectives we have developed the prototype system for the Honeynsand figure 1 shows the logical diagram of it. The following are the major modules of the honeysand system



5.1 Execution environment

Apart from acting as an environment for the malware execution it provides:

1. Confinement to the malware execution as it limits the effect of the malware execution to the virtual machine.
2. Provides quick system recovery.
3. Inbuilt data capture and collection mechanism

The sandbox environment is able to generate the required specifications in terms of operating system type, services running and software loaded dynamically. Every malware samples analyzed using honsysand have some meta data attached. This data is nothing but the system configuration specifications of the machine from where the sample was obtained.

This meta data is used by the honeysand at the time of the malware analysis to dynamically generate the similar system configuration for efficient execution of malware sample. In case if there is no prior information present regarding the malware sample the default configuration is used that is windows xp service pack 2 with normal services running.

Further the data capturing tools deployed in the execution environment captures the malware activity at two level of system abstraction. These are

- File system monitoring engine
- System API call hooking engine

Data captured by the file system monitoring tool and API call hooking tool is copied in to a hidden folder in the operating system and in the end of the analysis process a cross-time-diff technique based data collection system collects the complete data from these folders.

5.2 Traffic redirector

Traffic redirector is a very critical component of the whole analysis system. As it derives all the traffic generated by the malware sample in to the controlled network environment. The traffic redirector irrespective of the IP address diverts all the traffic generated by the malware sample towards network controller.

5.3 IP emulation engine

The IP emulator emulates the IP addresses it masquerades the destination IP address of any packet forwarded by the traffic redirector to it.

5.4 Protocol Handler

Handles the TCP, UDP, ICMP protocols.It has the capability to complete the three way TCP handshake, reply the ICMP echo packets, sends fake replies to show that given port or service is open or closed.

5.5 Service Proxy

Service proxy forwards the connection on a given port for a given service to the related fake server running on local network. It acts as a proxy between the malware and the server Hence by doing this it fools the malware to think that they are getting the reply from the original server where as in reality it coming from a common server hosted in-house.

5.6 Service Emulator

Service emulator is actually a set of scripts that emulates certain critical services such as SMTP, pop, telnet, and ftp. These scripts only provides the initial login banners and in some cases the command sets after showing the successful login that the user can use.All the commands usernames and passwords used to login by the users are logged in a log file. This provides us the ability to know profile the attacker and its techniques and motives.

5.7 Limited Internet

The HTTP is the most critical protocol to handle because of the fact it is common to have HTTP traffic and all most all the organizations have it but on the other hand this protocol is also used by the bot malware for.

1. Checking the genuineness of the internet connectivity provided to them. For this they try to connect with some legitimate internet websites and if is able to connect then it further proceeds.
2. Egg downloading various malware samples does use HTTP protocol for downloading the egg binary which actually performs the BOT behavior. Hence until allowed to download the binary samples from the internet the researcher will not be able to analyses the real bot behavior.
3. Communicating with the C&C sever. The HTTP bots uses the http traffic as a medium to communicate with their C&C server.
4. Launching any attack. Due to above mentioned issues the HTTP protocol needs a special consideration. As in some cases selective HTTP traffic should be allowed and in some cases no traffic should be allowed. Hence keeping in view these points we have proposed a solution where the HTTP traffic is passed to the HTTP proxy and at the HTTP proxy server the user can see, manipulate, drop, stop all the http traffic to and from the internet and malware sample.

5.8 Fake Victims

A combination of a low and high interaction honeypots are used in the honeysand design. These honeypots acts as the fake victim and provides the malware some real machines to attacks. The attack traffic for the well known vulnerabilities is redirected towards these honeypots. They provide the information about the propagation strategy of the malware sample and the information regarding the vulnerability that is exploited and how it is exploited.

5.9 Server Response Manipulation Scripts

Every malware sample is observed for some limited amount of time called window period. In some cases it becomes necessary to accelerate the malware executions to get the desired behavior triggered in the window period. For example in our bot analysis case many bots use to resolve the domain name of the C&C server if resolved successfully it further proceeds by logging in to the server. In some cases the domain names may not get resolved may be because of the busting of the botnet by security agencies in those cases the malware ceases to show further activities as it keeps sending the DNS requests and nothing else.To handle situations like these we have proposed the use of the manipulation scripts that actually manipulate the response packets for the malware sample based upon certain conditions. As for the proof of concept we have created such a script of the DNS server. The DNS request created by the malware is forwarded to the DNS handler script this script first tries to resolve the domain name if the domain name is successfully resolved the resolved IP address is sent to the malware but if the domain name could not be resolved successfully the scripts manipulates the response by creating a fake response packet with an random IP address that's been emulated by the sandnet.The moment the malware sample receives the response it further proceeds by initiating the TCP three way handshake which is also performed successfully by the protocol handler module. Once

the connection is established the malware sends the login and password credentials to the emulate IP address which are captured by the honeysand.

5.10 Real services

Honaysand runs certain servers locally which cater the different service needs of the malware samples. Initially we are running a time server for UDP based time stamp queries generated by the malware sample. The service proxy engine identifies the request for the 123 port and forwards it to the local time server and later it gets the response from the timeserver and sends it to the execution environment.

5.11 Honeysand base engine

Honeysand base engine controls and coordinates all the activities of the malware analysis.

- Creating the execution environment and virtual network between different system entities
- Extraction and processing of network and system logs.
- Fetching the malware sample from the database

The idea behind the creation of honeysand is to have a system with a diverse range of network solutions for addressing the network issue related to the Bot malware analysis.

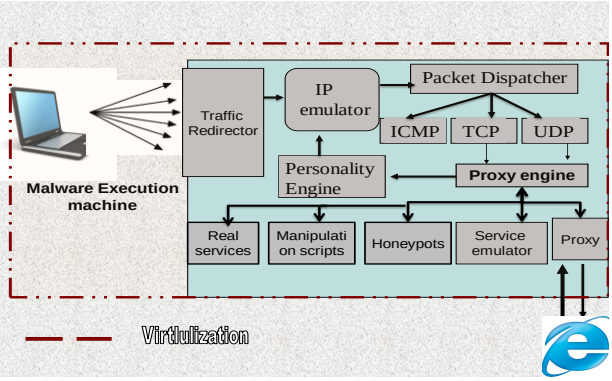


Figure 2

The prototype system hence developed is shown in figure 2.For the development of this system as per the above design specification and making it a cost effective solution we have used open source tools only. The use of the virtualization has greatly reduced the cost of the system as the whole setup could be created on a single machine in a virtual environment. The network diagram of the Honeysand system is shown in the figure 3.

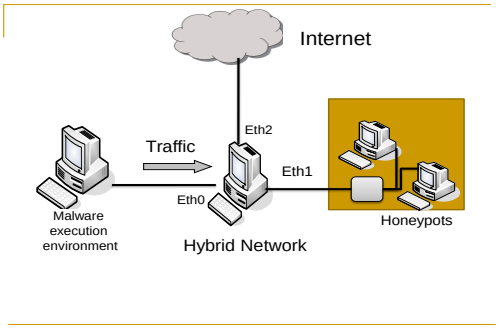


Figure 4 network Diagram

The malware sample is executed in the malware execution environment and the traffic from the execution environment to outside world is controlled by the Hybrid Network. The second machine named Hybrid Network has three network interfaces one directly connected to the internet other is connected to the execution environment and the third one is connected to a network of high and low interaction Honeypots.

Further figure 5 gives the more detailed view of the system in terms of the open source tools that have been used to realize the conceptual design mentioned in figure 1.

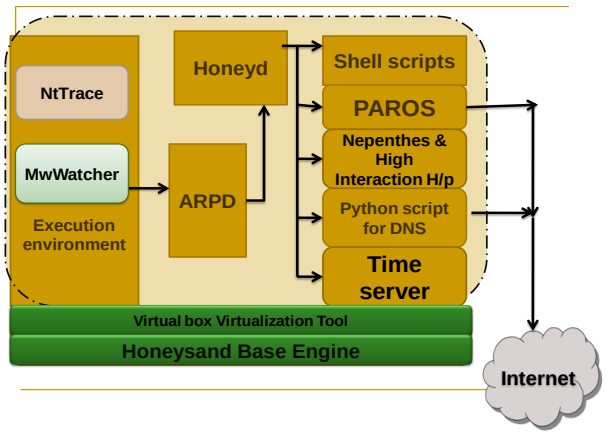


Figure 5 System Prototype

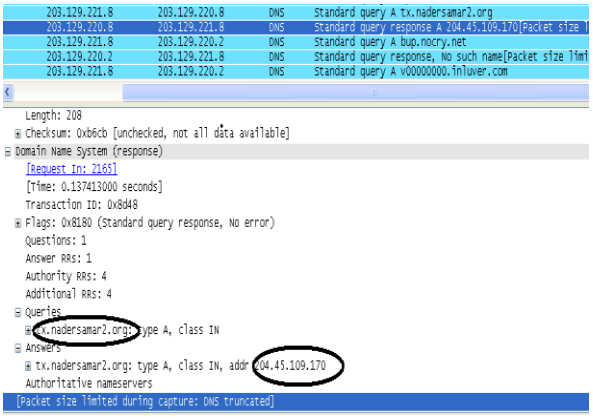
List of various open source tools used in the development of the prototype system.

- ARPD arp spoofing tool used to redirect all the network traffic generated by the malware execution environment towards the honeyd.
- Nttrace: is freely available windows Ntaive API call tracing software. It is used to track the malware execution at system level.
- Mwwatcher: is for monitoring the file system change
- Honeyd: is a low interaction honeypot it forms the core of the Honeysand.Infact the whole system is based upon this low interaction hoenypot.It provides the following functionality
- Emulation of the IP space
- Handling the protocols
- Acting as an Proxy engine
- Logging and data capturing
- TCPdump captures and saves the network traffic in PCAP format
- TCPFLOW Extracts and reassembles the TCP payloads
- Shell scripts: emulate the services some of the shell scripts comes as a part of the package honeyd,even one can write his own shell scripts to emulate different services.

- Paros: is an HTTP proxy that gives the capability to capture, edit, and drop the HTTP request and response packets.
- Honeypots: act as the default targets the traffic for known exploits is redirected to the honeypots.
- Manipulation script: manipulate the server response to accelerate the malware execution.
- Time server: a real time server as a service is running for queries of port 123.

6. RESULTS

To demonstrate the effectiveness of the honeysand we have performed certain experiments by executing the malware sample and observing the response of the system.



In the figure 6 the initial network traffic from the execution of IRC worm is shown.

1. The figure 6 shows that the bot is first resolving a domain name. The request is handled by the proxy service which forwards it to DNS manipulation script. DNS manipulation script forwards the request to a real domain name server. The domain name was successfully resolved and hence the DNS manipulation script returns the resolved IP address. Had it been an unsuccessful response the DNS manipulation script would have sent a fake IP address.

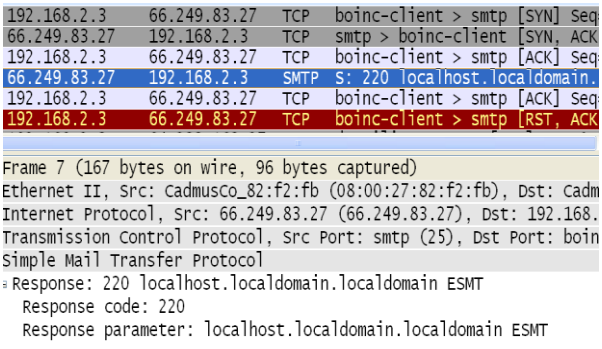


Figure 7

2. Once the domain name is resolved the malware sends the TCP syn packet for a TCP connection initiation. This is shown in the network traffic in figure 7.This TCP syn packet is handled by the honeyd as it emulates all internet space hence the request for TCP syn is received by the honeyd which further performs the three way handshake on behalf of the requested IP for initiating the connection
3. Once the connection is established the malware directly sends the login credentials to the server as shown in figure 8

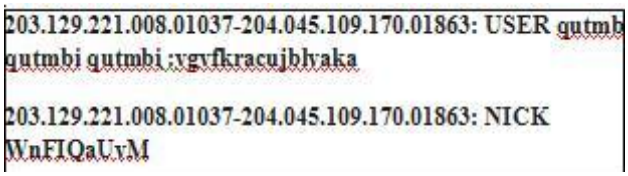


Figure 8

In another case in which an HTTP bot was analyzed following sequence of events occurred.

1. The malware sample resolved the Domain name

Figure 9

The figure 9 shows the logs from the Python based DNS manipulation script. The first domain name wlgtdqqgsec.cc was not been successfully resolved hence the default IP "203.128.223.22" was sent by the DNS manipulation script in response. Based upon this reply the server initiated the TCP connection which was done by the honeyd and then sends the following HTTP GET request.

GET / HTTP/1.1

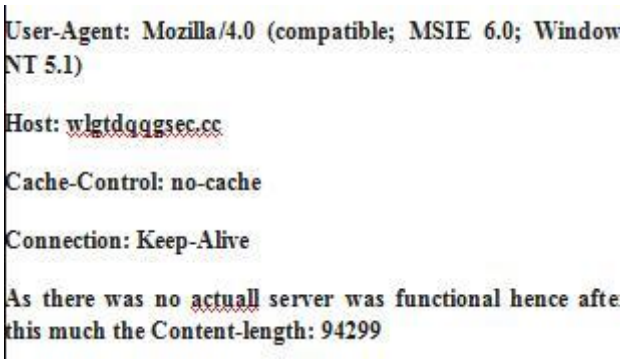


Figure 10

1. As there is no real HTTP server running at the fake IP hence after waiting for some time the malware has closed this connection and sent a request to resolve another domain name. This domain name was been successfully resolved.

2. Once the domain name is resolved the malware sends the same HTTP GET request. This request first goes to paros the HTTP proxy. As this was been just a file download request hence we allowed it and the reply was the file contents sent by the http server. Figure 11 shows the traces of this communication. This is a common instance of egg download prosess followed by the bot malwares.

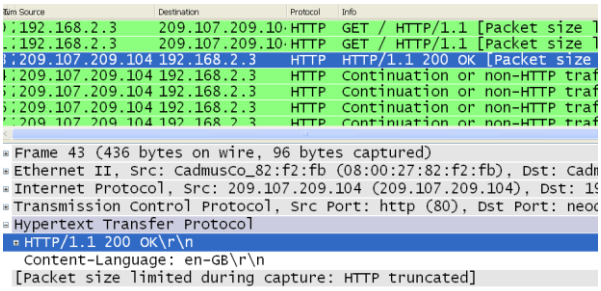


Figure 11

Further for the tracking of the botnets related to the detected bots we use the domain names. As every bot binary tries to resolves a set of domain names which are actually the C&C server or the egg download sites we use the domain name clustering technique. Figure 12 shows such a list of domain names and the sanitized IP addresses of the C&C servers.

C&C Servers IP

DNS	IP
italian.swiifatecihno.com	69.4 61.153
tx.nadersamar2.org	204 109.170
tx.nadersamar2.org	204 109.170
bbs.moiservice.com	64.7 23.39
tx.nadersamar2.org,everydns.net	204 109.170
ihshsd8.com|ns4.above.com|abc.ihshsd8.com	69.4 61.151
is-a-fag.net|esxt.is-a-fag.net	66.2 49.83
proxim.ircgalaxy.pl|ircgalaxy.pl|nss4.ircgalaxy.pl	83.1 119.197
tx.nadersamar2.org	204 109.170
tx.mostaffaljaafari.net|tx.nadersamar2.org	204 109.170

Figure 12

By applying clustering on the domain names we get a common set of domain names belonging to a Botnet.Any malware trying to resolve any domain lying in to this cluster will belongs to the same botnet.This is how we are tracking the botnets.Based upon this techniques we have tracked some of the botnets, Figure 13 shows a sample data set of botnets the domain name set and the bot samples that belongs to that botnet.

BOTNET	DOMAIN NAMES	Binary samples
N2	tx.nadersamar2.org, everydns.net, tx.mostaffaljaafari.net, mx1.hotmail.com, ftp.scarlet.be, mailin-01.mx.aol.com, tx.mostaffaljaafari.net, xx.enterhere.biz, yutunrz.1dumb.com, xx.sqlteam.info	b69,b68, b61, b2, b1, b112,b148, b74, b75, b90, b121, b125, b131, b148, b151
N8	soft.jajaca.com, in.jajaca.com,	b1010, b1022, b1036, b1071, b1013,1023,1039
N13	wanmei13722.3322.org	b1029, b1047
N14	aisinin1314.3322.or	b1048,b1072
N15	java.KUTLUFAMILY.COM	b1049, b1055, b1064, b114
N16	zhoucheng.3322.org	b1050, b1067

Figure 13

7. CONCLUSION & FUTURE WORK

The honeysand is a sandbox environment dedicated to the special requirements of the bot malware analysis. It provides an efficient cost-effective solution for the analysis of the bots. The information provided by it in the form of resolved domain names, IP addresses, username, passwords and the commands executed by the bot binary are quite helpful for the characterization of bots and tracking of the botnets. Some of the unique ideas introduced in this paper are

1. The use of the low interaction honeypot Honeyd as a core of this hybrid network
2. The use of the server response manipulation scripts for the accelerating the malware execution.

As most of the analysis performed in the hoenysand is dependent on the network traces hence the system may face certain difficulties in handling of the malware samples that uses the encrypted or covert channels for the communication.

Also the limited internet connectivity for the HTTP traffic relies completely on the expertise of researcher hence a second level of data control in the form of reverse firewall with IPS system is required for this

Dynamic malware techniques incorporates an overhead of malware execution and are non exhaustive. Further issues such as Multi threaded code, Kernel mode codes [14] are still a problem for dynamic malware analysis and needs to be addresses in the future work.

8. REFERENCES

[1] anubis.iseclab.org

[2] www.cwsandbox.org

[3] www.norman.com

[4] www.joebox.org

[5] netscty.com/malware-tool

[6] Shinsuke miwa,Toshiyuki miyiachi, Masashi eto, Masashi "Design and Implementation of an Isolated Sandbox with Mimetic Internet used to Analyze Malwares"

[7] Gérard Wagener • Radu State • Alexandre Dulaunoy "Malware behaviour analysis" Spinger-Verlag France 2007

[8] Andreas Moser, Christopher Kruegel, and Engin Kirda. "Limits of Static Analysis for Malware Detection". In

Proceedings of the 23rd Annual Computer Security Applications Conference(ACSAC), 2007

[9] Min Gyung Kang, Pongsin Poosankam, and Heng Yin. Renovo: "A Hidden Code Extractor for Packed Executables". In Proceedings of the 5th ACM Workshop on Recurring Malcode (WORM), 2007.

[10] Lorenzo Martignoni, Mihai Christodorescu, and Somesh Jha. "OmniUnpack: Fast, Generic,and Safe Unpacking of Malware. In Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC), 2007.

[11] Boris Lau and Vanja Svajcer. Measuring virtual machine detection in malware using DSD tracer. Journal in Computer Virology, 6(3), 2010.

[12] Thomas Raffetseder, Christopher Kruegel, and Engin Kirda "Detecting System Emulators"

[13] Vasudevan, Yerraballi,"Cobra: Fine-grained Malware Analysis using Stealth Localized-Executions". In: IEEE Symposium on Security and Privacy. (2006)

[14] Bayer, Kruegel,Kirda, "TTAnalyze: A Tool for Analyzing Malware".In: 15th Annual Conference of the European Institute for Computer Antivirus Research (EICAR). (2006)

[15] Xu Chen, Jon Andersen, Zhuoqing Morley Mao, Michael Bailey, and Jose Nazario. "Towards an Understanding of Anti-Virtualization and Anti-Debugging Behavior in Modern Malware". In Proceedings of the 38th Annual IEEE International Conference on Dependable Systems and Networks (DSN), 2008.

[16] Katsunari Yoshioka, Yoshihiko Hosobuchi, Tatsunori Orii, and Tsutomu Matsumoto. "Your Sandbox is blinded: Impact of Decoy Injection to Public Malware Analysis Systems". Journal of Information Processing, 19, 2011.

[17] Manuel,EgeleTheodoor,ScholteEngine Survey on Automated Dynamic Malware Analysis Techniques and Tools"

[18] Honeynet Project & Research Alliance. "Know your Enemy: Tracking Botnets".

[19] MichaelBailey,JonOberheide,JonAndersen,Z. Morley Mao, Farnam Jahanian,Jose Nazario. "Automated Classification and Analysis of Internet Malware"

[20] T. Dullien and R. Rolles. "Graph-based comparison of Executable Objects". In Symposium sur la S´ecurit´e des Technologies de l'Information et des Communications (SSTIC), June 2005.

[21] M. Christodorescu and S. Jha. Static "Analysis of Executables to Detect Malicious Patterns". In Usenix Security Symposium, 2003.

[22] C. Linn and S. Debray. "Obfuscation of executable code to improve resistance to static disassembly". In CCS '03: Proceedings of the 10th ACM conference on Computer and communications security, pages 290–299, New York, NY, USA, 2003. ACM.

[23] F. Guo, P. Ferrie, and T.-C. Chiueh." A study of the packer problem and its solutions". In RAID '08: Proceedings of the 11th international symposium on Recent Advances in Intrusion Detection, pages 98–115, Berlin, Heidelberg, 2008. Springer-Verlag.

[24] M. Christodorescu, S. Jha, S. Seshia, D. Song, and R. Bryant. "Semantics-aware Malware Detection". In IEEE Symposium on Security and Privacy, 2005.

[25] J. Kinder, S. Katzenbeisser, C. Schallhart, and H. Veith. "Detecting Malicious Code by Model Checking". In Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), 2005.

[26] C. Kruegel, W. Robertson, and G. Vigna. "Detecting Kernel-Level Rootkits through Binary Analysis". In Annual Computer Security Application Conference (ACSAC), 2004.

[27] http://en.wikipedia.org/wiki/Botnet