ISSN No. 0976-5697

Volume 3, No. 7, Nov-Dec 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

An Approach to Conceal Honeypot from Attacker

Mr. Dharmendra G. Bhatti*
Associate Professor
S. R. Institute of Management and Computer Application,
Bardoli, Gujarat, India
dgbhatti@yahoo.com

Ms. Sonal Bhakta
S. R. Institute of Management and Computer Application,
Bardoli, Gujarat, India
sonalbhakta@ymail.com

Ms. Ankita Shah
S. R. Institute of Management and Computer Application,
Bardoli, Gujarat, India
apshah_08@yahoo.co.in

Abstract: Honeypots are designed to attract the attackers and gather their information. It can be used to log an attacker's activities, analyze its behaviour and design new approaches to defend against it. As there is no human user working on the honeypot, there is absence of physical and network activity on it. This can easily provide the identity of the honeypot to the attackers. In order to operate effectively, it is required to conceal the honeypot from the attacker. Thus the attacker will attack on the host without knowing that it is a honeypot, and honeypot can collect more information about the attacker. For concealing the honeypot, here we present honeypot as a normal host by sending dummy files over the network. Generally honeypot does not contain any network traffic. So here we will generate network traffic around honeypot by using dummy files.

Keywords: Conceal Honeypot, Network Security

I. INTRODUCTION

General definition of honeypot is an information system resource whose value lies in illegal/unauthorized use of that resource. A honeypot is not designated as a production-oriented component of an information infrastructure. Ideally, none should be accessing honeypot; any interactions with a honeypot are by definition unauthorized. Thus honeypot identifies intruder and strengthen network security [1][2][3].

Honeypots can serve differently according to different situations. Honeypots can be categorized into two forms: low-interaction and high-interaction, as per the level of interaction between attacker and a honeypot. Low-interaction honeypots are simulated in such a way so that they cannot be exploited for getting complete access to the honeypot by the attackers. While high-interaction honeypots are design in such a way so that the attackers can get maximum access to the honeypots.

Honeypots can also be classified according to how honeypots are implemented. A honeypot with its own OS and IP address is a real honeypot means a physical honeypot, and a honeypot with imaginary IP address and emulated system is a virtual honeypot [4][5].

The advantages of honeypots are as follows:

a. Fewer false positives since no legitimate traffic uses honeypot
b. Collect smaller, higher-value, datasets since they only log illegitimate activity
c. Work in encrypted environments
d. Do not require known attack signatures

There are some disadvantages of honeypots as listed below:

a. Can be used by attacker to attack other systems
b. Only monitor interactions made directly with the honeypot - the honeypot cannot detect attacks against other systems
c. They can potentially be detected by the attacker

There is a lack of physical and network activity on the honeypots as there are no real human user works on honeypots. Because of so the identity of the honeypot can easily be revealed and once an identity of the honeypot is revealed, the attacker can avoid attacking over that honeypot so that the honeypot cannot gather information about that attacker. Therefore, in order to detect the attacker and gather the information about the attackers we need to conceal honeypot from the attacker.

In this paper, we have proposed a model to conceal honeypot from the attackers since the attackers can identify the honeypot so that they can avoid honeypot while attacking. In related work, various solutions to conceal honeypot, which are till now exist, are discussed. In proposed model, our proposed model for the solution of the problem of concealing honeypot from the attackers is presented and discussed.

II. RELATED WORK

Ng, Jun Ping [6] says that, as there is no real human user working on honeypots, so there is a lack of physical and network activity on the honeypots. This will potentially give-away of the honeypots' identity to the attackers. So the Honeypot have an autonomous mouse movements and keystrokes. This will convince an attacker observing the Honeypot that a human user is working on the machine, and lead them to believe that this is a real production system.

Nandan Garg and Daniel Grosu [7], says that when a host is probed, it gives certain response. This response is valuable

to the attacker as they can gain information about the honeypots. One way a defender can avoid the mapping of honeypots is to cleverly manipulate the response to the probe and deceive the attacker.

They propose a game theoretic model considering two players, the Attacker and the Defender (honeynet: contains number of honeypots). The response of the attacker depends on whether the host is a Honeypot or a regular host. The response of the Honeypot may be generated such that to conceal the identity of the Honeypot. But after a few probes an attacker can successfully identify that a host is a Honeypot i.e. the defender cannot infinitely deceive the attacker in believing that Honeypot is a regular host.

Xinwen Fu, Bryan Graham, Dan Cheng, Riccardo Bettati, and Wei Zhao [8], says that honeypot implementation fails due to the low fidelity of emulation of the system components by the honeypot and is emulated by timers in the honeypot. When the timers are carelessly defined in the honeypot implementation or are provided at insufficient accuracy by the underlying OS, a timing signature emerges. An attacker may construct a profile of a honeypot and launch a timing attack.

To camouflage honeyd and defeat the type of timing attacks, the modified honeyd and underlying OS will support for a high-fidelity emulation events. They have provided (a) accurately configured link latencies, and (b) high-resolution timers within the OS to solve this problem.

Sherif M. Khattab, Chatree Sangpachatanaruk, Daniel Moss´e, Rami Melhem and Taieb Znati [9], says that Honeypot are generally deployed at fixed location and on machines other than the ones they are supposed to protect, sophisticated attacks can avoid the honeypots. They propose a roaming honeypots, a mechanism that allows the locations of honeypots to be unpredictable and continuously changing. A (continuously changing) subset of the servers is active and providing service, while the rest of the server pool is idle and acting as honeypots.

The benefits of roaming honeypots scheme are: Firstly, idle servers detect attacker addresses so that all their subsequent requests are filtered out, which is known as filtering effect. Secondly, each time a server switches from idle to active; it drops all its current connections, opening a window opportunity for legitimate requests before the attack re-builds up, which is known as the connection dropping effect. Whereas the filtering effect defends the service against attacks launched from outside a firewall (external attacks), the connection-dropping effect mitigates attacks launched from behind the firewall (internal attacks).

Neil C. Rowe, Han C. Goh [10], says that one of the best ways to defend a computer system is to make attackers think it is not worth attacking. Deception provides a large variety of specific tactics that can confuse, scare away, or tie up an attacker depending on the circumstances and the methods. Honeypot can try to disguise their monitoring activities by concealing their monitoring software and monitoring messages.

The sebek tool of the honeynet project is a kind of "defensive rootkit". It conceals its monitoring process by rewriting the process-listing utility of the OS to omit it, and conceals its monitoring code by modifying the OS directory-listing utility. Deception can delay or halt suspicious activity when implemented as false error messages, demand on the attackers, or outright stalling.

Asmund Nergard Nyre [11], says that the increased dependence on computer systems to provide support for critical services, calls for additional measures to guarantee the continued deliverance of services even under attack. He proposed a system capable of tolerating attacks, while preserving the integrity, confidentiality and availability of the system to its legitimate users.

Upon detecting an attack, the compromised server is relieved from active duty and dynamically transformed into a state of Honeypot, while letting the attacker retain control of the server. By not alarming the attacker of the detection, they are left wasting their time exploiting the honeypot, providing system administrators with useful information in the subsequent patching of the security hole.

S. Antonatos, M. Athanatos, J. Velegrakis, N. Hatzibodozies, S. Ioannidis, E. P. Markatos, K. G. Anagnostakis [12], says that it is not difficult for the attackers to identify honeypots and develop blacklists to avoid them when launching an attack. The Tor assumes that every sender knows the address of the recipient. But here the address of the Honeypot should remain hidden.

Tor offers "hidden services", a functionality that permits to hide the address of the recipient. In hidden services the recipient gets a descriptor for its hidden service from a centralized service lookup server. Afterward, it creates onion paths to several introduction points. An introduction point can be any onion router. It then advertises the descriptors of introduction points and addresses to service lookup repository. The clients only need to know the service descriptor.

Nathan Willis [13], says that attackers can be trips in two ways: First, to slows them down by vastly increasing the amount of work they must do to correctly identify the real target machines on the network. The more you slow down the attacker, the better you get a chances of catch them through other methods. Second, no legitimate user on your network will ever need to probe a Honeyd virtual server, because they do not offer real services. Therefore any probes or connection attempts are automatically red flags.

Many researchers have suggested different ideas for improvement like decoy document [14], honeystat [15], virtual honeypot [16][17][18], helpful to IDS [19], packet fragmentation [20], botnet identification [21], assessing effect [22], Java based honeypot[23]. Few other researchers [24][25][26][27] have discussed improvements in architecture and process. Special cases like spim honeypot [28], hybrid honeypot [29], honeypot detection countermeasures [30], and deception limitations [31] are also addressed by researchers.

III. PROPOSED MODEL

For concealing honeypot from attackers, the proposed model consists of mainly two components:

a. Traffic Generator
b. Traffic Identifier

The general identity of the honeypot is that, only the attackers are attacking on the honeypot. So based on the server

traffic an attacker can distinguish whether the server is honeypot or a normal server. Honeypot is having the least network traffic. Thus the traffic generator component will send the request to the server to generate the network traffic on honeypot. The request will be generated at the random time interval. So when the attacker tries to detect that whether the server is a honeypot or not, they will not be able to identify the honeypot.

As to conceal the identity of honeypot; we are generating dummy traffic. So it is required that we should differentiate the dummy request from the actual attacker's request. Thus to differentiate the dummy request from the actual request we proposed a component, which is traffic identifier. The traffic identifier will differentiate the actual request from the dummy request based on the sequence number set by the traffic generator as well its IP address.

The sequence number will be sent to the server with the packet. Before sending the request, the sequence number will be set in encrypted format. The sequence number should be encrypted because if an attacker may get the sequence number it will not be able to get the original sequence number and thus can't get honeypot identity.

A. Traffic Generator:

The traffic generator component will work to generate the dummy traffic over the honeypot. So the traffic generator component will be installed on the client machines over the network. Thus each client machine will send a request to the honeypot server at random time interval using the traffic generator.

The traffic generator component gets the current system time and sets the curr_time and req_time. The req_time is the time at which the dummy request has been sent.

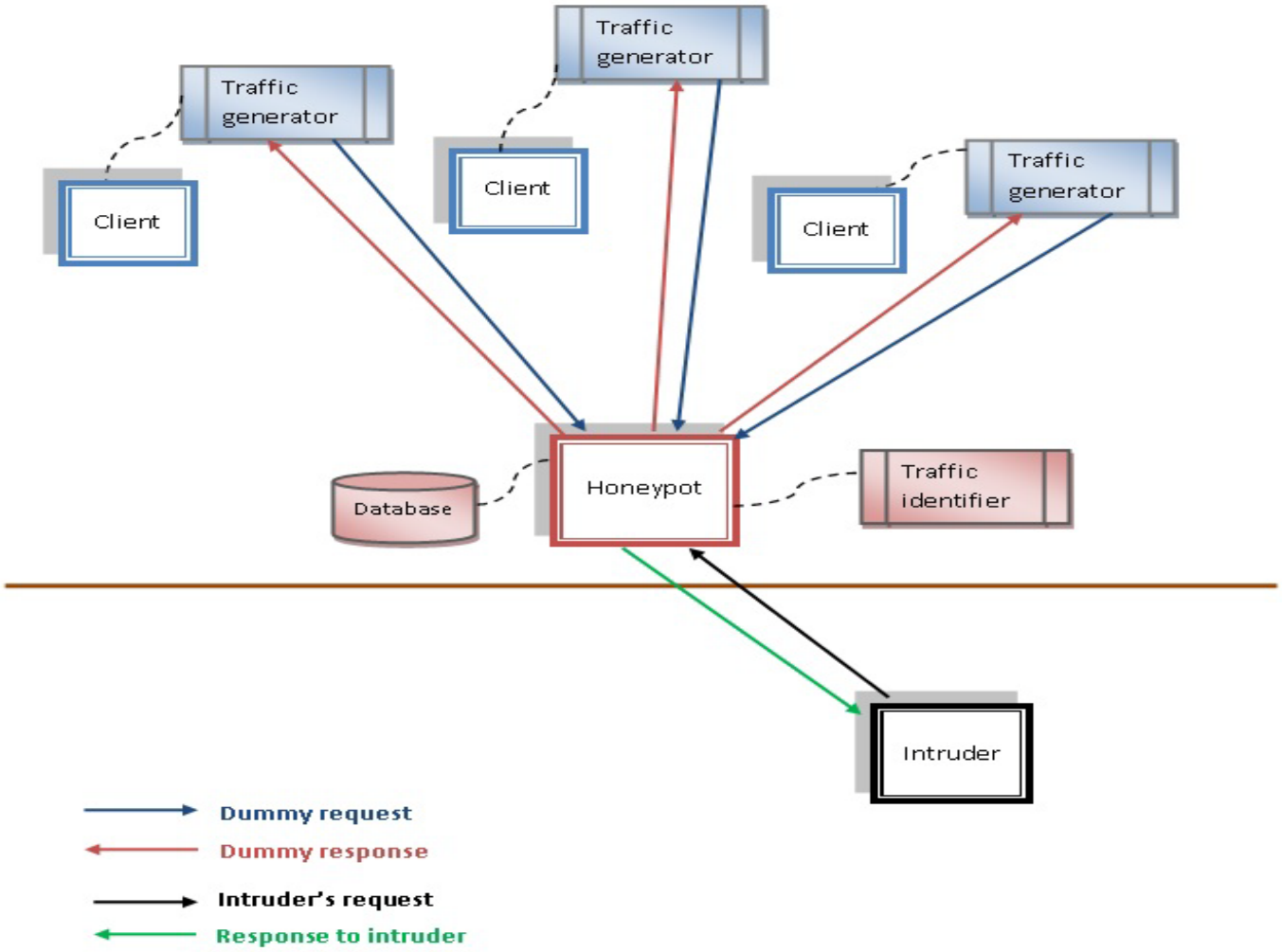

Figure 1. Proposed Model of Concealed Honeypot

Now to send the request at random time interval, a random number is generated and set as interval_time. So after every interval_time, next request is been sent to the honeypot server by the traffic generator component.

We need to differentiate the dummy request from the attacker request; thus by adding the encrypted sequence number in packet we can recognize the dummy request. So a seq_number in encrypted form is set in the packet and a request is sent over honeypot server.

After sending request, traffic generator will receive response with encrypted seq_number in packet. The received seq_number is the result of the incrimination of sent

seq_number, i.e. if client has sent n as seq_number, it will receive n+1 as seq_number in response.

For sending the next request, the curr_time is compared with the req_time and interval_time. When the curr_time exceeds the req_time + interval_time, the next request will be sent to the honeypot server.

B. Traffic Identifier:

The traffic identifier will differentiate the dummy request from the actual attacker's request. So the traffic identifier component will be installed at the server side, which is on the honeypot server.

Traffic identifier will retrieve the information from the received request. It will retrieve the client's IP address and other packet information like seq_number. From the packet, it will retrieve the seq_number and decrypt that seq_number.

After getting all the information it will search for an IP address in the database and verifies the seq_number with the expected seq_number according to the previous seq_number sent to the client.

If this information is valid then it means that the user is known client and request is the dummy request generated by the traffic generator component. Thus an appropriate response, with incremented seq_number stored in packet, is sent back to the client.

But if the sender is an attacker then an appropriate response is sent to the intruder. Also the honeypot will collect all the other information related to the attacker and will store it in appropriate manner.

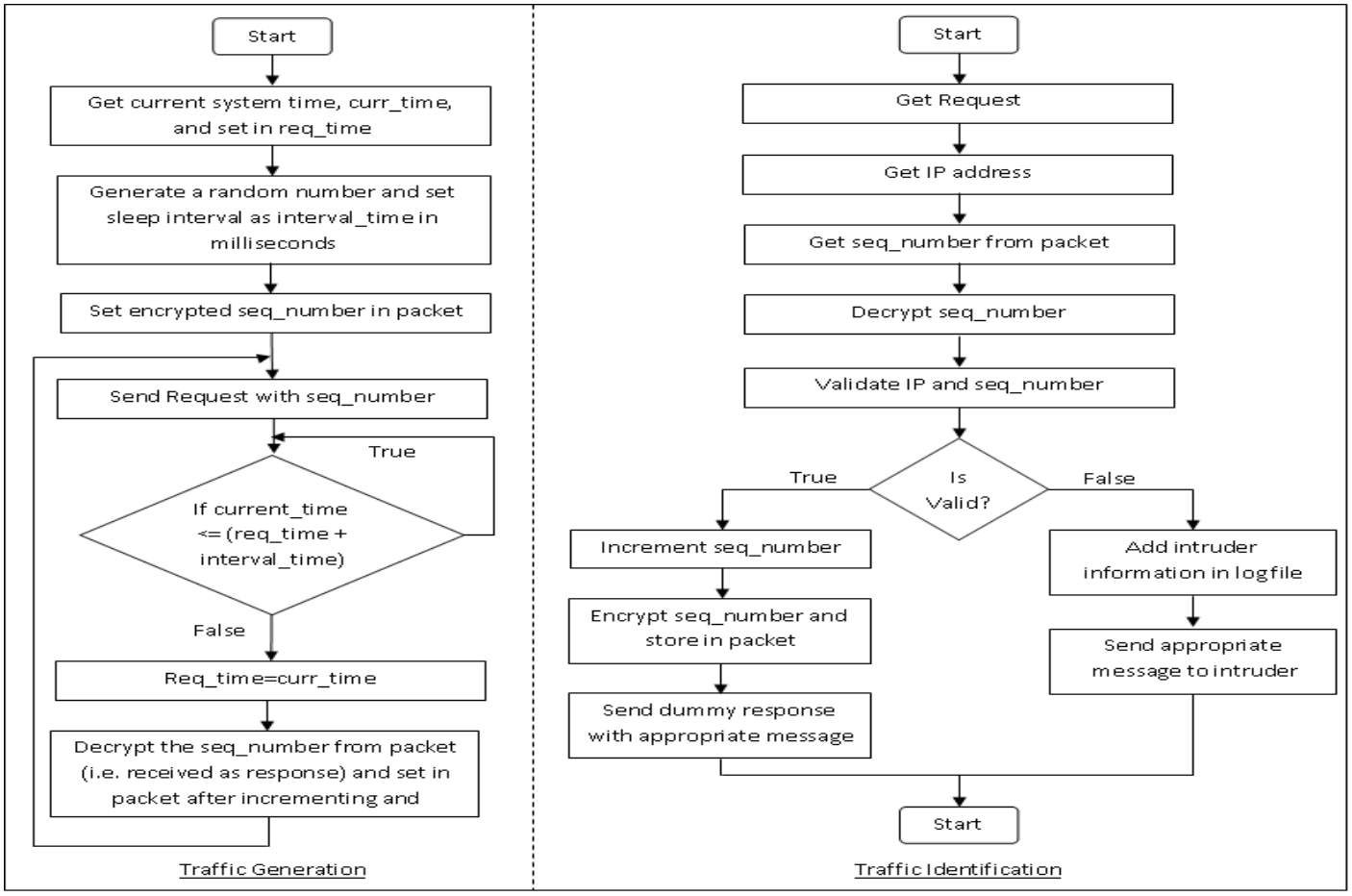

Figure 2. Traffic Generation and Identification Steps

C. Database:

The database contains the log file consisting the request and response details. It may possible that the attackers also keep an eye on the information transferred during request and response.

So for concealing, an employee database is maintained with basic information. Thus information is stored and accessed during request-response so that intruder cannot discover honeypot.

IV. RESULT ANALYSYS

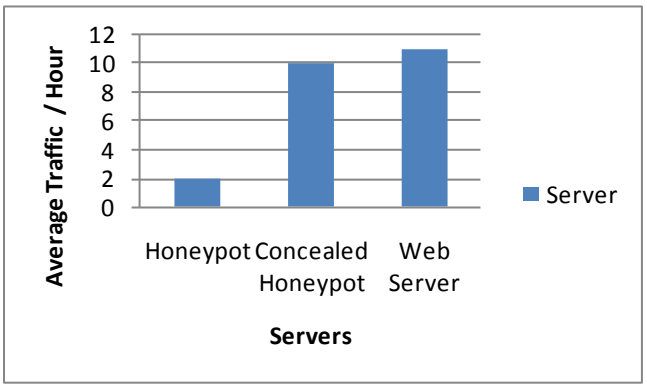

Figure 3. Average traffic

Figure-3 shows an average traffic rate for normal honeypot, concealed honeypot, and web server. From this chart we can say that there is a significant difference between the average traffic rate per hour over the web server and honeypot which is not concealed. Because of so, an intruder can detect honeypot since there is no traffic over the honeypot as compared to the web server. While in case of concealed honeypot, there is no significant difference between average traffic rate over the concealed honeypot and the web server.

Figure-4 shows the traffic rate of concealed honeypot and web server at particular time. Here, we can see that there is no significant difference between traffic over the concealed honeypot and web server. Thus if the traffic over the concealed honeypot and/or over the web server is analyzed then the traffic over the concealed honeypot will be found similar as over the web server and so concealed honeypot will not be found as the honeypot and will be assumed as web server by the intruders.

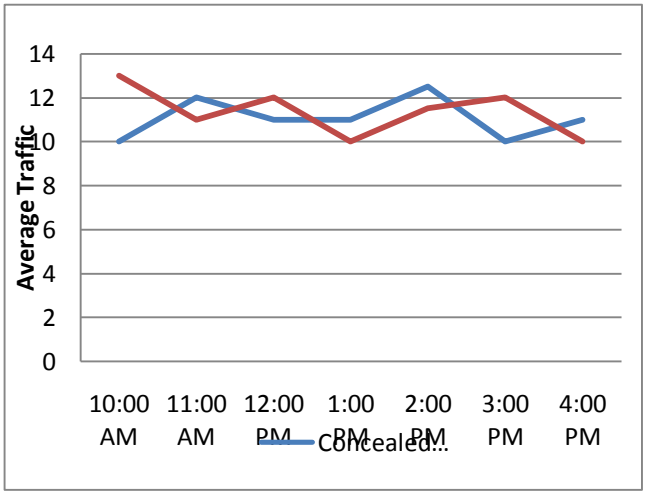

Figure: 4 Concealed Honeypot v/s Web Server

V. CONCLUSION

Generally honeypot is not accessed by client over the network. It is just installed to gather the information about the intruders. Unless the intruder does not directly attack over the honeypot, honeypot is not able to recognize the intruder. So there is no traffic over the honeypot. Thus an intruder can easily identify a honeypot by scanning over its network connection. So for concealing the honeypot, we are creating the dummy traffic over the honeypot. This traffic will present honeypot as a webserver and so the identity of the honeypot will be concealed from the intruder. As dummy traffic is generated, we need to differentiate it from the intruder's attack. Thus by implementing the traffic identifier we can differentiate the dummy traffic from the intruder's attack.

VI. REFERENCES

[1] Feng Zhang, Shijie Zhou. Zhiguang Qin, Jinde Liu, "Honeypot: a Supplemented Active Defense System for Network Security", Proceedings of the Fourth International Conference Parallel and Distributed Computing, Applications and Technologies on 27-29 August 2003 [Print ISBN:0-7803-7840-7, pp 231-235].

[2] Jim Yuill, Dorothy Denning, and Fred Feer, "Using Deception to Hide Things from Hackers: Processes, Principles, and Techniques", The Journal of Information Warfare, Volume 5, Issue 3, November 2006.

[3] Neil C. Rowe, "Deception in Defense of Computer Systems from Cyber-Attack," chapter 13 in L. Janczewski and A. Colarik, Cyber Warfare and Cyber Terrorism, pp. 97-104, 2007.

[4] K. G. Anagnostakis, S. Sidiroglou, P. Akritidis, K. Xinidis, E. Markatos, A. D. Keromytis, "Detecting Targeted Attacks Using Shadow Honeypots", 14th Conference on USENIX Security Symposium – Volume 14, doi=10.1.1.124.8215, July 2005. (Article in a conference proceedings)

[5] Xinwen Fu, Wei Yu, Dan Cheng, Xuejun Tan, and Steve Graham, "On Recognizing Virtual Honeypots and Countermeasures, Dependable, Autonomic and Secure computing", 2nd IEEE International Symposium on 29/9/2006-1/10/2006, Print ISBN: 0-7695-2539-3. (Article in a conference proceedings)

[6] Ng, Jun Ping, "Enhancing Honeypot Stealthiness", Thesis of Master of Science in Computer Science at Singapore-MIT Alliance, June 2006, Unpublished.

http://www.comp.nus.edu.sg/~junping/docs/njp-msc-thesis.pdf

[7] Nadan Garg and Daniel Grosu, "Deception in Honeynets: A Game-Theoretic Analysis", Proceedings of the 2007 IEEE Workshop on Information Assurance, Print ISBN:1-4244-1304-4, United States Military Academy, West Point, NY, 20-22 June 2007. (Article in a conference proceedings)

[8] Xinwen Fu, Bryan Graham, Dan Cheng, Riccardo Bettati, and Wei Zhao, "Camouflaging Virtual Honeypots", Technical Report- Department of computer Science, Texas A&M University, College station, TX 77843 – 3112, 2005, Unpublished.

[9] Sherif M. Khattab, Chatree Sangpachatanaruk, Daniel Moss´e, Rami Melhem, Taieb Znati, "Distributed Computing Systems", 2004. Proceedings. 24th International Conference, 2004, ISSN:1063-6927, Print ISBN:0-7695-2086-3. (Article in a conference proceedings)

[10] Neil C. Rowe, Han C. Goh, "Thwarting Cyber Attack Reconnaissance with Inconsistency and Deception", Information Assurance and Security Workshop, 2007. IEEE SMC, 20-22 June 2007, Print ISBN:1-4244-1304-4. (Article in a conference proceedings)

[11] Asmund Nergard Nyre, "Increasing Survivability by Dynamic Deployment of Honeypots", Master's Thesis, Norwegian University of Science and Technology, Faculty of Information Technology, Mathematics, and Electrical Engineering, April 2005, Unpublished.

http://www.sislab.no/NYRE_mastersthesis_printerfriendly.pdf

[12] S. Antonatos, M. Athanatos, J. Velegrakis, N. Hatzibodozies, S. Ioannidis, E. P. Markatos, K. G. Anagnostakis, "Hooney@home:A New Approach to Large-Scale Threat Monitoring", Information Security Threats Data Collection and Sharing, 2008. WISTDCS '08. WOMBAT Workshop on 21-22 April 2008, Print ISBN:978-0-7695-3347-6. (Article in a conference proceedings)

[13] Nathan Willis, "Weekend Project: Use HoneyD on Linux to Fool Attackers", 2011, Unpublished.

https://www.linux.com/learn/tutorials/472795:weekend-project-use-honeyd-on-linux-to-fool-attackers

[14] Brian M. Bowen, Shlomo Hershkop, Angelos D. Keromytis, Salvatore J. Stolfo, "Baiting Inside Attackers Using Decoy Documents", SecureComm 2009, 5th International ICST Conference on Security and Privacy in Communication Networks, September 14-17,2009, Athens, Greece, doi=10.1.1.150.1361, 2009. (Article in a conference proceedings)

[15] David Dagon, Xinzhou Qin, Guofei Gu, Wenke Lee,Julian Grizzard, John Levine, and Henry Owen, "HoneyStat: Local Worm Detection Using Honeypots", 7th International Symposium on Recent Advances in Intrusion Detection Conference, Sophia Antipolis, France, doi=10.1.1.87.2299, September 2004. (Article in a conference proceedings)

[16] Guillaume Chazarain, Benoˆıt Vallette d'Osia, Nicolas Nobelis, Karima Boudaoud, "A virtual high-interaction Honeypot", I3S Laboratory, University of Nice Sophia Antipolis, France, doi=10.1.1.97.4769, 2008, Unpublished.

[17] Xuxian Jiang, Dongyan Xu, Collapsar: "A VM-Based Architecture for Network Attack Detention Center", In Proceedings of the 13th USENIX Security Symposium, Volume 13. (Article in a conference proceedings)

[18] Xuxian Jiang, Dongyan Xu, Yi-Min Wang, "Collapsar: A VM-Based Honeyfarm and Reverse Honeyfarm Architecture for Network Attack Capture and Detention", CERIAS and Department of Computer Science, Purdue University, West Lafayette IN 47907, Miscrosoft Research, Redmond, WA 98052, Available online 16 July 2012 http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2643&context=cstech

[19] Jako Fritz, "Hybrid Intrusion Detection Network Monitoring with Honeypots", Master's thesis, University of Twente, 27 April 2011, Unpublished.

http://essay.utwente.nl/60830/1/MSc_J_Fritz.pdf

[20] Jon Oberheide, Manish Karir, "Honeyd Detection via Packet Fragmentation", Networking Research and Development, Merit Network Inc., January 2006, Unpublished.

www.merit.edu/networkresearch/papers/pdf/2006/MTR-2006-01.pdf

[21] Ping Wang, Lei Wu, Ryan Cunningham, Cliff C. Zou, "Honeypot Detection in Advanced Botnet Attacks", International Journal of Information and Computer Security, Volume:4, Issue:1, ISSN online: 1744-1773, ISSN print: 1744-1765, 2010

[22] Sze Li Harry Lim, "Assessing The Effect Of Honeypots On Cyber-Attackers", Thesis- Master Of Science in Computer Science from Naval postgraduate School, December-2006, Unpublished.

[23] V. Maheshwari, P. E. Sankaranarayanan, "Defeating Hackers Through a Java Based Honeypot Deployment", Information Technology Journal, Volume:6, Issue:7, 2007

[24] P. Diebold, A. Hess, G. Schafer, "A Honeypot Architecture for Detecting and Analyzing Unknown Network Attacks", Conference proceeding of Kommunikation in Verteilten Systemen (KiVS), 14. ITG/GI-Fachtagung Kommunikation in Verteilten Systemen (KiVS 2005) Kaiserslautern, 28. Februar - 3. März 2005. (Article in a conference proceedings)

[25] Jim Yuill, Dorothy Denning, and Fred Feer, "Using Deception to Hide Things from Hackers: Processes, Principles, and Techniques", The Journal of Information Warfare, Volume 5, Issue 3, November 2006.

[26] Shubham Gupta, Vishal Singhal, "HONEYPOT: A Trap for Hackers", Proceedings of the 5th National Conference; INDIACom-2011,Computing For Nation Development, March 10 – 11, 2011. (Article in a conference proceedings)

[27] Suen Yek, "Implementing network defence using deception in a wireless Honeypot", Proceeding of 2nd Australian Computer, Network & Information Forensics Conference 2004: Perth, Western Australia, 2004. (Article in a conference proceedings)

[28] Aarjav J. Trivedi, Paul Q. Judge, Sven Krasser, "Analyzing Network and Content Characteristics of Spim using Honeypots", SRUTI'07, Proceedings of the 3rd USENIX workshop on Steps to reducing unwanted traffic on the internet, doi=10.1.1.81.7693, 2007. (Article in a conference proceedings)

[29] Kyi Lin Lin Kyaw, "Hybrid Honeypot System for Network Security", World Academy of Science, Engineering and Technology, Volume 24, No, 44, 2008, Unpublished.

[30] Lai-Ming Shiue, Shang-Juh Kao, "Countermeasure for Detection of Honeypot Deployment", Proceeding of Internation Conference on Computer and Communication Engineering, 2008. ICCCE 2008, Print ISBN : 978-1-4244-1691-2. (Article in a conference proceedings)

[31] Maximillian Dornseif, Thorsten Holz, und Sven Müller, "Honeypots and Limitations of Deception", Lecture Notes in Informatics Proceedings, Volume P-73, No. 14, 2005

http://subs.emis.de/LNI/Proceedings/Proceedings73/GI-Proceedings.73-14.pdf