International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:12, 2008

Hybrid Honeypot System for Network Security

Kyi Lin Lin Kyaw, Department of Engineering Physics, Mandalay Technological University, Pathein Gyi, Mandalay.

Abstract—Nowadays, we are facing with network threats that cause enormous damage to the Internet community day by day. In this situation, more and more people try to prevent their network security using some traditional mechanisms including firewall, Intrusion Detection System, etc. Among them honeypot is a versatile tool for a security practitioner, of course, they are tools that are meant to be attacked or interacted with to more information about attackers, their motives and tools. In this paper, we will describe usefulness of low-interaction honeypot and high-interaction honeypot and comparison between them. And then we propose hybrid honeypot architecture that combines low and high -interaction honeypot to mitigate the drawback. In this architecture, low-interaction honeypot is used as a traffic filter. Activities like port scanning can be effectively detected by low-interaction honeypot and stop there. Traffic that cannot be handled by low-interaction honeypot is handed over to high-interaction honeypot. In this case, low-interaction honeypot is used as proxy whereas high-interaction honeypot offers the optimal level realism. To prevent the high-interaction honeypot from infections, containment environment (VMware) is used.

Keywords—Low-interaction honeypot, High-interaction honeypot, VMware, Proxy

I. INTRODUCTION

These days, network security is threatened by the attacker, and we need to prevent viruses and worms from penetrating the network with the use of patches and security update software. But no network is impenetrable in time; the network security shall be compromised.

Those who integrate honeypots into their network take a different approach. The purposes of honeypot are to detected and learn from attacks and use that information provides network security. Honeypots are analyzed by their role of application, which is meant it can be used for production and research. To describe them in greater detail, it is necessary to explain the level of interaction with the attacker. This paper is organized as followed. In section 2, we describe definition of honeypot. In section 3, we explain characterizes of low-interaction honeypot, high-interaction honeypot and comparison between them. In section4, we describe community between low-interaction honeypot and high-interaction honeypot , its application and conclusion are discussed in section5.

II. DEFINITIONS OF HONEYPOT

According to Lance Spitner, author of Honeypots, Tracking Hackers [4],

"A honeypot is security resource whose value lies in being probed, attacked, or compromised".

A honeypot is a system that is built and set up in order to be hacked. Honeypot can be used in a different scenario as intrusion detection facility (burglar alarm), defense or response mechanism. Moreover, Honeypot can be deployed in order to consume the resources of the attacker or distract him from the valuable targets and slow him down that wastes his time on the honeypot instead of attacking production systems [1]. The main functions of a honeypot are (Pouget & Holz, 2005) [9]:

- to divert the attention of the attacker from the real network, in a way that the main information resources are not compromised
- to capture new viruses or worms for future study
- to build attacker profiles in order to identify their preferred attack methods, similar to criminal profiles used by law enforcement agencies in order to identify a criminal's *modus operandi*
- to identify new vulnerabilities and risks of various operating systems, environments and programs which are not thoroughly identified at the moment.

III. LEVEL INTERACTION OF HONEYPOT

The level of interaction is defined as the range of attack possibilities that a honeypot allow an attacker to have, where as it can be classified as high- interaction honeypot and low-interaction honeypot.

A. High- Interaction Honeypot

In high- interaction honeypot, attacker interaction with real operating systems, services and programs and it can be used to observe the attackers behavior, their tools, motivation and explored vulnerabilities. This kind of honeypot must have a robust containment mechanism in order to prevent, once compromised, its use to attack other networks. One goal of a hacker is to gain root and to have access to a machine, which is connected to the internet 24/7. A high- interaction honeypot does offer such an environment. To facilitate the deployment of machines, automatic installation through images retrieved from system from Quattor can be used. Tools like Sebek can help high-interaction honeypot to instrument to log and/or system calls.

A high- interaction honeypot can be installed inside a virtual machine using virtualization software such as VMware, Qemu and Xen. Using virtualization software, the attacker may run specialized code to detect that his code is running inside a virtual machine environment or perform timing attacks to identify honeypots. And performance of applications running in the guest operating system is reduced. However, an effort is made in the architecture to reduce the load of high-interaction honeypots by preprocessing the traffic using low-interaction honeypots as much as possible. Example of high-interaction honeypot is honeynet. A honeynet is a network of multiple systems. Honeynet [6] can collect in-depth information about attackers, such as their keystrokes when they compromise a system, their chat sessions with fellow black hats, or the tools they use to probe and exploit vulnerable systems. This data can provide incredible insight on the attacker themselves. The advantage with honeynet is that they collect information based on the attackers' actions in the wild.

B. Low- Interaction Honeypot

On low- interaction honeypot [3], there is no operating system that an attacker can operate on. Tools are installed in order to emulate operating systems and services. And they interact with the attackers and malicious code. This will minimize the risk significantly. This kind of honeypot has a small chance of being compromised. It is production honeypot. Typical use of low-interaction honeypot includes; port scans identification, generation of attack signatures, trend analysis and malware collection.

On the other hand, this is also a disadvantage [2]. It is not possible to watch an attacker interacting with the operating system, which could be really interacting. Example of low-interaction honeypot is honeyd. Honeyd is an open source low-interactivity honeypot system that creats virtual hosts that can be configured to run arbitrary services and their personality can be adapted so that they appear to be running certain operating systems. Honeyd [3] - [8], enables a single host to claim multiple addresses. Honeyd improves cyber security by providing mechanism for threat detection and assessment. It also deters adversaries by hiding systems in the middle of virtual systems.

It is possible to ping the virtual machines or to trace out them. Any type of service on the virtual machine can be simulated according to a simple configuration file. Instead of simulating service, it is also possible to proxy it to another machine. A complete picture of how honeyd work is shown in following.

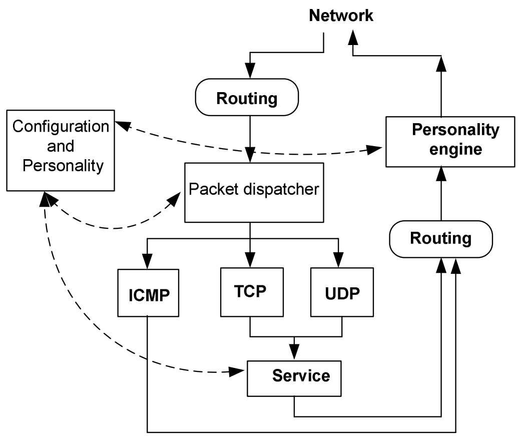


Fig. 2: A simplified view of the honeyd architecture

The IP stack is emulated in user space and packets are delivered to TCP, UDP or ICMP handler is the components that perform TCP stack emulation. Generally, handlers forward packets to the appropriate services emulated by scripts. Packets with destination port 80 are e.g. handed over to the web server emulation script. The scripts can be either external programs or proxies to real services. The personality engine is the one that is responsible for setting up a behavior for an emulated IP address. As described before, honeyd can emulate various operating systems at network level.

C. Comparison between low- interaction honeypot and high-interaction honeypot

Each level has advantages and disadvantage as mention below;

II. TABLE 1: SUMMARIZES OF LOW AND HIGH-INTERACTION HONEYPOT

	Low-interaction honeypot	High-interaction honeypot
Degree of Involvement	Low	High
Real Operating System	No	Yes
Risk	Low	High
Information Gathering	Connections	All
Compromised Wished	No	Yes
Knowledge to Run	Low	High
Knowledge to Develop	Low	Mid-High
Maintenance Time	Low	Very High

IV. HYBRID HONEYPOT ARCHITECTURE

In our experiment, both low- interaction honeypot and high-interaction honeypot are deployed Also low-interaction honeypot is more secure than high- interaction honeypot because of running real service; it lacks the ability to provide a good level of realism[7]. However, high- interaction honeypot provides the best possible level of realism but it has more risk. To provide an extensible infrastructure for honeypot deployment and development of detection mechanisms on top, good properties of both types have to be combined.

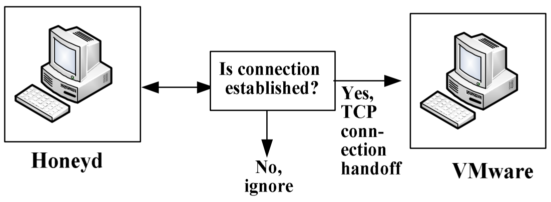


Fig. 3:Hand off and filtering mechanism

In this system, low- interaction honeypot act as lightweight proxy [3]. We want high-interaction honeypot to process all traffic destined to black IP address space. We need to offload them as front end to high-interaction honeypot because it is instrumented machines. Honeyd has the appropriate properties to play the role of the front end and acts as a filtering component. The lightweight proxy responds only to TCP/SYN requests to ports that are open. For any other ports, it just absorbs and records the packets received. When the three-way handshake has completed properly between the attacker and the low- interaction honeypot, the connection must be hand-off to the appropriate high-interaction honeypot . At this point, also referred as zero point, the low-interaction honeypot set as a connection with the high- interaction honeypot. The low-interaction honeypot sets as like relay agent. Any application level data coming from attacker is forwarded to the high-interaction honeypot and vice versa, until the connection is terminated. This behavior is embedded to the honeyd implementation, know as proxy mode. The proxy mode is instrumented to record the message exchanges, for further analysis purposes. Hand-off is useful in case of port scanning, where low-interaction honeypot will absorb all incoming connections without disturbing high-interaction honeypot.

In the following illustration, initially, the attacker sends a TCP/ SYN packet to the low-interaction honeypot. If the honeypot is configured to listen to the port, then it sends a SYN/ACK packet and waits to receive the next packet. If the packet is not an ACK then the low-interaction honeypot assumes that it was a port scan and the connection is dropped. If the third packet received is ACK then it is a valid TCP connection and the zero point is reached. Thus the low-interaction honeypot connects with the high-interaction honeypot running the requested service. Then after the connection establishment the low-interaction honeypot continues to work as a proxy. As low and high- interaction honeypots belong to the same local network, no additional delay will be perceived by the attacker.

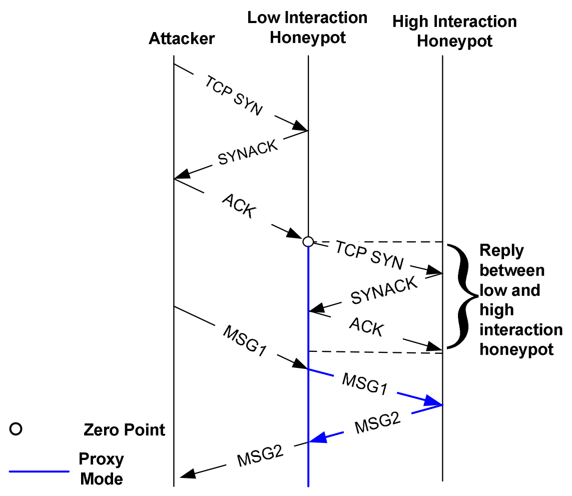


Figure 4: An example of connection hand-off

A. Application of the hybrid framework

In this section, we discuss the applicability of this framework that included distributed lightweight proxy deployment, a centralized VMware system, and the connection handoff to solving some of the standard problems in Internet thread detection and resolution.

B. Detection

One unique application of the hybrid framework is to the area of worm detection. Recall that the backend component of actual hosts serving as honeypots. Our novel worm propagation detection mechanism actually watches for the propagation of worms. Worms often use the same propagation method from host to host; we can apply the same content check summing algorithm to packet out of the backend honeypot and match them to the MD5 of the inbound connection. A matching outbound signature that matches an inbound handoff signature is even a higher indicator of self propagation code.

Here, we give criteria that characterize the methodology that was applied to evaluation attack detection quality [11]. The following criteria characterize the methodology used to evaluate attack detection:

Attack Detection Setup: The setup used to evaluate attack detection. Parameters can be either:
-real-world deployment: The system was deployed in a productive network.
– lab deployment, real-world attacks: The role of the attacker is assigned to a host/user in the lab environment. We use real world attacks.
– lab deployment, synthesized attacks: The role of the attacker is assigned to a host/user in the lab environment. We use synthetic attacks.
– lab deployment, replayed attacks: Replay of real-world traffic traces.
This includes modifications such as sanitization and attack injection.
• Measured Parameter. It can be either:
– FP: False Positives
– FN: False Negatives
– TP: True Positives
– TN: True Negatives
• Probe Size: The number of different attacks (true positives/false negatives) or the number of benign interactions (true negatives/false positives) if available.
• Establishment of Ground Truth. Parameters can be either:
– manual analysis: We did a manual analysis to find the actual number of attacks.
– reference system: The number of identified attacks was compared to the number of attacks found by a reference system.
– attack injection: A specific number of attacks was injected.
– attack only: All interactions with the system belonged to attacks. Hence, this allows true positive/false negative analysis only.
– benign only: All interactions with the system were benign.
Hence, this allows true negative/false positive analysis only.

– concept: The applied attack detection concept guarantees a minimum/maximum value for the measured parameter(s).

C. Signature Quality

The following criteria characterize the methodology used to evaluate signature quality:
• Signature Generation Setup: The setup used to generate the set of signatures for the signature quality assessment. Parameters can be either:
– real-world deployment: The system was deployed in a real-world network environment while generating the set of signatures.
– lab deployment, synthesized traffic: The traffic used to generate the signature set was either hand-crafted or dynamically generated using a tested.
– lab deployment, replayed traffic traces: The traffic used to generate the signature set was generated by replaying a set of (real-world) traffic traces.
– lab deployment, replayed but modified traffic traces: A set of (partially) modified (real-world) traffic traces was replayed.
– analytically assessed: No experimental setup. The signature quality was assessed analytically.

If a system generates e.g. a signature that identifies allowed/approved actions/behavior, an evaluation of its signature generation mechanism is actually done when evaluating the attack detection mechanism. Further criteria to characterize the methodology used to evaluate signature quality are:

Quality Assessment Setup. Parameters are the same as for the Signature Generation Setup. If not stated otherwise, the network traffic for the assessment is not identical to the network traffic used to generate the set of signatures.

• Measured Parameter. It can be either:
– FP: False Negatives
– FN: False Positives
– TP: True Positives
– TN: True Negatives
• Establishment of Ground Truth. Parameters can be either:
– manual analysis: Either the authors analyzed the network traffic manually to find the number of attacks contained in it or they analyzed if the total set of traffic that the signature/filter describes/filters contains only attacks.
– reference system: Either the number of identified attacks was compared to the number of attacks found by a reference system or the attacks reported by the reference system were removed to ”sanitize” the traces.
– attack injection: A specific number of attacks was injected into network traffic.
– attack only: The network traffic consisted of attacks only.

D. Signature generation

Signature generation is the process of defining all the necessary characteristics of a new thread to be able to detect a new occurrence of the threat, identify existing infected hosts, and immunize against additional infections. This process is uniquely suited to the hybrid architecture as is required, sufficient number of mentioned hosts to catch the threat early in its growth phase and sufficient detailed behavioral analysis to identify previously unseen threats.

We propose to use a combination of host- and network-based attack detection algorithms, namely Dynamic Taint Analysis and Root Cause Analysis. The interaction between both systems is depicted in Figure 5. Traffic which is directed to the high interaction honeypot will be split and redirected to a root cause analysis engine. This engine monitors the attack activity on the network level, and generates an alert in case a new root cause is detected. For deciding whether a root cause is novel or not, a central database will be contacted. In a second step the alert correlator matches the DTA and RCA alerts. Alert correlation can be done based on simple timing information, i.e. when was an alert triggered. However, since the alert generation time can differ significantly for RCA and DTA, it would be better to correlate only alerts that were generated by the same network traffic. This is especially important in case a sensor is attacked very frequently. Therefore, it is necessary that the DTA system is able to identify the network traffic that was responsible for a generated alert.

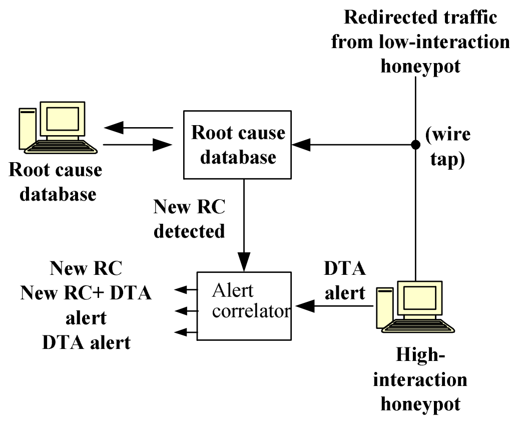


Fig: 5 Signature Generations

E. True/ False Positive Ratio

True Positive Ratio (TPR) is a way showing how good the intrusion detection is at alerting on real attacks. In our setting we use this to better performance. TPR is obtained by the following formula:

$$TPR = \frac{TP}{TP + FN}$$

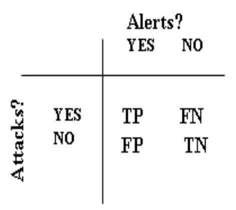
Where, TP= The number of alerts on malicious traffic, FN= The number of missing alerts on malicious traffic. The total number of intrusion is given by TP+FN.. False Positive Ratio(FPR) shows the proportion of instances, which were not an attack but still were alerted on. FPR is result of the following formula:

$$FPR = \frac{FP}{FP + TN}$$

Where, FP=The number of alerts on benign traffic, TN= The number of correct decisions on benign traffic. The total number of no-intrusion is given by FP+TN.

A perfect system would have TPR=1 and FPR=0. This would result in alerts only on malicious traffic, and no alerts on benign traffic.

The confusion matrix in figure 6 illustrates what FP, FN, TP and TN mean.



		Alerts? YES	Alerts? NO
Attacks?	YES	TP	FN
	NO	FP	TN

Figure 6: Confusion matrix

F. Essence of Hybrid honeypot

High-interaction honeypot	Low-interaction honeypot	Hybrid honeypot
- Slow	+ Fast	+Fast
+ Able to detect unknown attacks + 0 False positive	- Unable to detect unknown attacks	+ Able to detect unknown attacks + 0 False positive
- Unable to deal with time bombs and user interaction	+ Able to deal with time bombs and user interaction	+ Able to deal with time bombs and user interaction
- Expensive	+ Cheap	+ Better RoI
- Difficult to setup and operate	+ Easy to setup and operate	- Difficult to setup and operate

V. Conclusion

Using hybrid honeypot, we achieve a number of goals. First, we need to maintain only a small number of high- interaction honeypots since the portion of the traffic will be routed to them is limited. All port- scan attempts or connection to port that is not open will be stopped by low-interaction honeypots. Second, the high-interaction honeypots will be placed in a monitored network. Thus if a honeypot gets infected, the infection rate will be contrololable either through limiting bandwidth or traffic reflection. Also, since honeyd can emulate different machines running in a network, we can map several machines which run the same operating system and similar services to a single high-interaction honeypot. Finally, the addition of new services to the high- interaction is facilitated we only should open appropriate port at the low-interaction honeypot and set up the mapping.

Honeypots offer a unique perspective to defending networks by learning the habits and techniques of the blackhat at an additional cost of minimal network alert reporting and monitoring time.

REFERENCES

[1] P.Diebold,A. Hess, G,Schafer. A Honeypot Architecture for Detecting and Analyzing Unknown Network Attacks. In Proc. Oh 14th Kommunikationin Verteilten systemen 2005(KiVS05), Kaiserslautern, Germany, February 2005

[2] Honeypots: White Paper. Reto Baumann, http:// www. Rbaumann.net, Christian Plattner, http:// www. Christianplattner.net

[3] Research infrastructures action, Sixth framework programme, D1.1: Honeypot Node Architecture, page 7-24

[4] Spitzer, Lance. Honeypots, Tracking Hackers. Pdf version. Addison Wesely,2002.

[5] Spitzer, Lance. Honeypots- Definitions and Value of Honeypots. http://www.infosecwriters.com, March 6,2003.

[6] Honeynet project, The. (2007a). Know your enemy: Honeynets. Retrieved on 7 October 2007 from http;//www. Honeynet.org/papers/honeynet/index.html

[7] Research infrastructures action, Sixth framework programme, D1.4: Architecture Integration, page 36.

[8] Niels Provos: Honeyd- Virtual Honeypot, http://www.honeyd.org/, Provos 2002

[9] Pouget,F., & Holz, T. (2005). A pointillist approach for comparing honeypots. In K. Julisch & C. Kruegel (Eds), Intrusion and malware detection and vulnerability assessment. Berlin/ Heidelberg: Springer

[10] Tyad Kuwatly, Malek Sraj, Zaid Al Masri, A Dynamic Honeypot Design for Intrusion Detection, American U. of Beirut .2004.

[11] Research infrastructures action, Sixth framework programme, D1.2: Attack detection and signature generation

.