Using PFSense and Commodity Hardware as a Medium Interaction Honey-net

Georgios Chlapoutakis[1], Anastasios Laskos[2], Phillip J. Brooke[3], Mark Truran[4]

Abstract

Honey-pots and *honey-nets* are network-accessible decoy resources designed to attract unauthorized users, thereby deflecting their attention from security-critical systems. Combined with a process known as a LaBrea *tar pit*, honey-net and honey-pots can effectively entrap suspicious connection attempts and prevent the proliferation of further attacks via the host network. Use of these 'sticky' honey-pots and honey-nets is quite common within the network security community, and several 'off the shelf' solutions are available to individuals and commercial clients alike.

In this paper we describe a *LaBrea* tar-pit honey-net solution specifically designed for researchers interested in network security. Unlike the various honey net solutions mentioned above, this solution gives the user direct access to *raw, packet-level data*. Our reference implementation is built around a customized firewall distribution which acts as the *honey-net bridge*. This bridge uses an BSD-based firewall distribution known as *PFSense* in combination with a highly customizable network packet capturing facility called *tcpdump*.

The contribution of this work is twofold. Firstly, our reference implementation will enable researchers to quickly deploy a medium interaction honey-net network (with LaBrea Tar-pit functionality) that is more secure, more adaptable and more upgradeable than comparable systems. Secondly, our reference implementation will allow researchers to transparently gather raw, live-traffic data without the additional overhead of performing on-site traffic analysis. It is hoped that this will stimulate higher quality dataset collection throughout the network security field.

[1] School of Computing, Teesside University, United Kingdom, g.chlapoutakis@tees.ac.uk

[2] School of Computing, University of Sunderland, United Kingdom, bd74yy@student.sunderland.ac.uk

[3] School of Computing, Teesside University, United Kingdom, pjb@scm.tees.ac.uk

[4] School of Computing, Teesside University, United Kingdom, m.a.truran@tees.ac.uk

1. Introduction

Network security and digital forensic science researchers and practitioners interested in network attacks for purposes such as analysis, inference and profiling have, over the course of the last few years, repeatedly made use of computer systems, or entire networks of said systems, purposefully left vulnerable in order to study and gather data on network security attacks.

According to Spitzner and The Honeynet Project team (Spitzner 2003, The Honeynet Project 2005), honeypots and honeynets have been the source of much-needed data used in the advancement of methodologies such as anomaly detection in intrusion detection systems, or more recently theories of network attack profiling.

Current approaches on honeynet and honeypot technologies, as well as current data analysis and findings through the use of said technologies, almost exclusively rely on a very small number of tools or toolkits, such as the Honeywall (https://projects.honeynet.org/honeywall/) honeynet and the honeyd (http://www.honeyd.org/) and nepenthes (Nepenthes 2009) honeypots.

However, while said tools and toolkits have evolved over time to incorporate technologies to not only capture but also analyse and create reports on network attacks, thus creating an all-encompassing solution for the main-stream network security and digital forensic areas of research, they have also forced a number of restrictions on the practitioners of these areas.

An example of such restrictions can be considered the lack of raw and in-depth network traffic captures that can be gathered and safely stored for further off-line analysis through the use of software external to the honeynet-embedded ones. Another such example is the reliance on relatively large footprint honeynet platforms (upwards of 600MB disk space).

The contribution of this work is twofold. Firstly, our reference implementation will enable researchers to quickly deploy a medium interaction honey-net network (with LaBrea Tar-pit functionality) that is more secure, more adaptable and more upgradeable than comparable systems. Secondly, our reference implementation will allow researchers to transparently gather raw, live-traffic data without the additional overhead of performing on-site traffic analysis. It is hoped that this will stimulate higher quality dataset collection throughout the network security field.

Section 2 of this paper will focus on the research conducted on network attacks, from either a network security or digital forensic point of view, through the use of existing honeynet and honeypot solutions. In Section 3, we discuss the methodology our reference implementation follows for the purpose of data gathering and the number of restrictions our work had to abide by. In Section 4, the construction of the honeynet network, the honeypot machines, the packet capturing and the tar pit methodologies we used in our reference implementation will be detailed.

Section 5 will describe the testing process employed as well as some initial results of our analysis of the network data that was captured, and we will discuss the meaning and impact of our reference implementation, as well as further work in Section 6.

2. Related work

In his research paper on the value of such purposefully vulnerable systems as network security tools, Spitzner used the term honeypots as a way to define

> "…an information system resource whose value lies in unauthorized or illicit use of that resource." (Spitzner 2003)

Such systems, according to The Honeynet Project team (The Honeynet Project 2005) can either be low or high interaction. In a low interaction honeypot an operating system or service is emulated and thus offers the attacker little in the way of actual interaction with the system. A high interaction honeypot, however, real operating systems, services and applications are provided for the attacker to interact with.

High-interaction honeypot systems, when networked in order to emulate a vulnerable network, have become known as first, second and third-generation honeynet networks (The Honeynet Project 2005b, The Honeynet Project 2006). Each successive generation adds further depth and degrees of automation to the data gathering, analysis and reporting phases.

The most widely and extensively known and used low and high-interaction honeypot and honeynet solutions used in both network security and digital forensics have been the honeyd Unix daemon (Provos 2004) and nepenthes (Nepenthes 2009), as low-interaction honeypots. Also the Honeywall toolkit (The Honeynet Project 2005b) as a 2nd and 3rd generation honeynet solution.

2.1 Uses of Honeypots

The uses of honeypots and honeynets differ in their scope and depth between the fields of network security and digital forensics.

In the field of network security, honeypot and honeynet solutions have been most prevalent in the study of botnets and worms as well as in studies of common network attacks and Distributed Denial of Service attacks.

For example, Cooke et al. (2005) and later Zhuge et al. (2007) both used low-interaction honeypots to study the trends and behaviors of malicious scripts (malware) in trying to, and successfully infecting computer systems. Cooke et al used the simplest form of a medium-interaction honeypot and honeynet system, which is a number of systems purposefully left vulnerable with a transparent proxy bridge, while Zhuge et al. used the low-interaction honeypot toolkit nepenthes.

In order to study the propagation rates of Internet worms and devise appropriate zero-hour response methodologies to detect them, Portokalidis and Bos (2007) used a combination of high and low-interaction honeypots in combination with intrusion detection systems.

Kuwatly et al. (2004) and Krasser et al. (2005) both advocated the use of honeynet and honeypot networks in order to detect and model network intrusions, with the motive of raising the issue of the need for increased security awareness in both network administrators and network users.

Digital forensic science has made use of honeypots and honeynets as well, but mostly in order to gain knowledge on current and emergent network attacker tactics that can aid in further cases where such tactics are argued by the prosecution or defense in a court of law. Such is the case with Yasinsac and Manzano's (2003) research into the use of a combination of honeynets and honeypots, which they collectively named "honeytraps", to study attacker behavior in a compromised system.

Similarly with Ren and Jin's honeynet-based active network intrusion response system (Ren and Jin 2005), which revolved around using a high-interaction honeynet as an automated network data gathering and analysis module of what is otherwise a standard intrusion detection system, and Thonnard and Dacier's (2008) research which focused on using honeynets to gather data on network attackers, subsequently performing attack classification using clustering.

3. Requirements and methodology

Our choice of the reference implementation presented in this paper, as well as the methodology followed to arrive at it, over either Free/Libre/Open-Source Software (FLOSS) such as the aforementioned honeyd and Honeywall, or off-the-shelf solutions, were heavily influenced by a number of requirements.

An important requirement was that the reference implementation was built for the purpose of gathering data for a digital forensic attack and attacker profiling system. The Association of Chief Police Officers guidelines on the handling of computer-based evidence (ACPO 2007), which are discussed further in Marshall's book (Marshall 2008), require computer evidence to be unchanged whenever possible. This should also apply to network data, including the data packets captured. It could be argued that this data should ideally be complete , and be stored without either any changes or any pre-processing.

In that respect, the solutions offered by most honeypot and honeynet implementations (high and low-interaction alike) produce data of insufficient detail to satisfy the above requirement. We suggest that it should include network packet traces including full IP header information and full packet payload information.

With regards to the honeynet linux bridge, now, the main and overriding issue with most FLOSS solutions was that their implementation of a linux honeynet bridge was (and has been for a long time) stated as work in progress.

Other requirements and specifications that our reference implementation had to consider was the use of Network Address Translation in the construction of the honeynet network, so as to allow full access to the entities wishing to use the services advertised, at the same time denying the user of these services access to both the other computers in the internal network and the Internet in general through them.

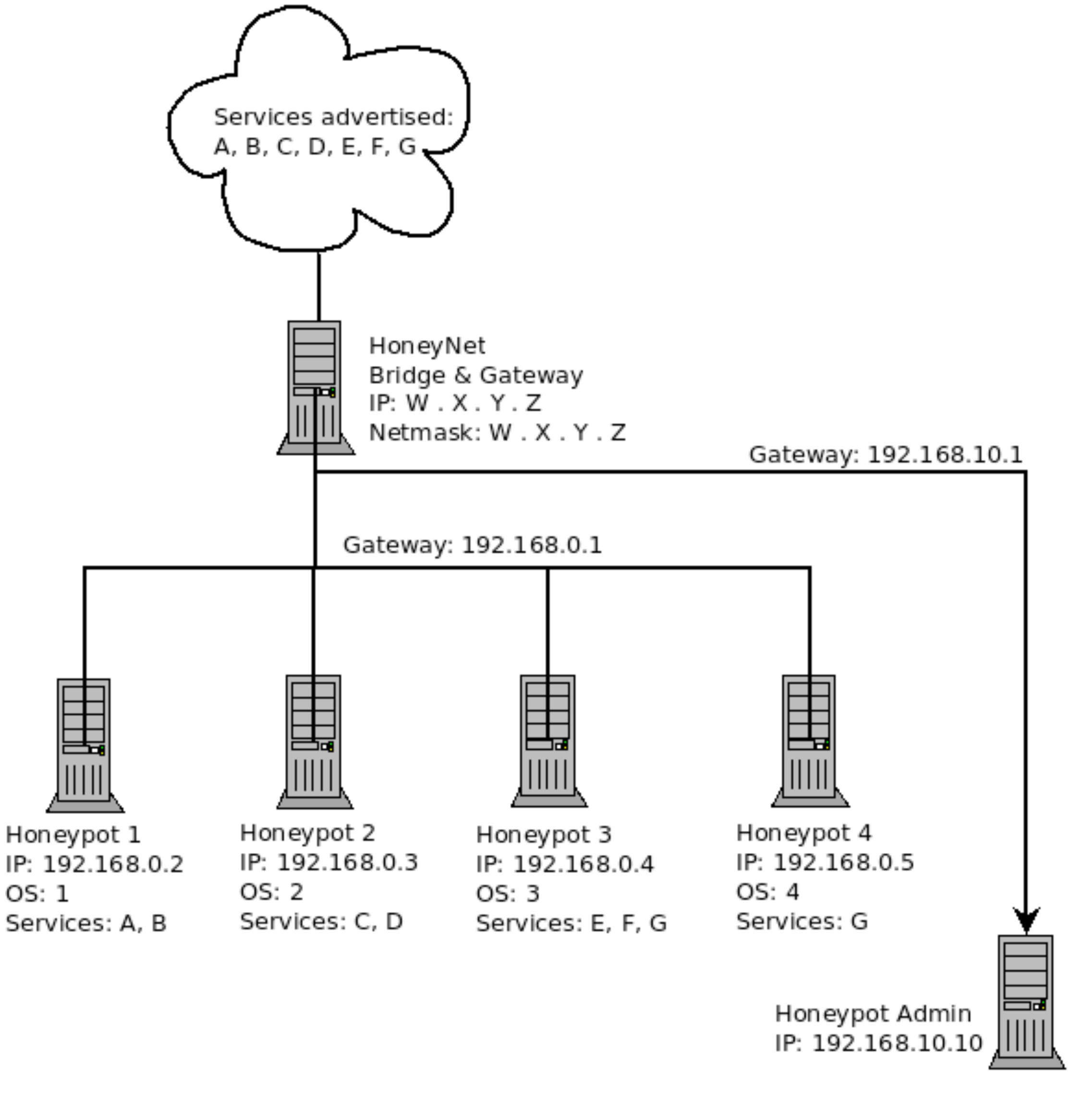

Figure 1: A rough view of the honeynet network

Figure 1 shows a rough view of the honeynet network and the honeypot computers. From the layout of the network, it can be seen that it was modeled to resemble as closely as possible a typical Small-to-Medium Enterprise (SME) Local Area Network, with the honeynet bridge & gateway being either a standard (and, to some extent, not very well configured) DSL router using NAT and port-forwarding a number of services to the rest of the world.

Finally, it was deemed necessary that the network traffic data be gathered through a network connection originating from the honeynet bridge to a computer using a dedicated IP address, and the data transfer to be encrypted (via sftp). The combination of the different interface and the secure file transfer mechanism was deemed to be sufficient to keep any attackers which have successfully penetrated the honeynet bridge from both identifying and altering the data transfer in progress.

The requirements and specifications detailed above were identified prior to the construction of the actual honeynet network and were the product of research into the issue of communications monitoring (packet sniffing) for the purposes of network security research., as detailed in Slicker et al. (2007), Pang et al. (2004) and Paxon (2004).

4. Construction of the Honeynet Network

The honeynet network is comprised of one honeynet bridge giving the outside world access to four honeypot computers, each of which running one or more services.

All of the machines acting as honeypots have older versions of known operating systems, or unpatched and insecure versions of current operating systems, or older and/or insecure well-known services. This is partly due to the fact that the purpose of a honeynet network is to be broken into, so strict security is not a necessity.

It also is partly due to our wish to emulate the real-world scenario of an SME computer network which is sometimes either unmanaged or not properly managed by an IT administration team and, thus, has a number of outdated operating systems and services running.

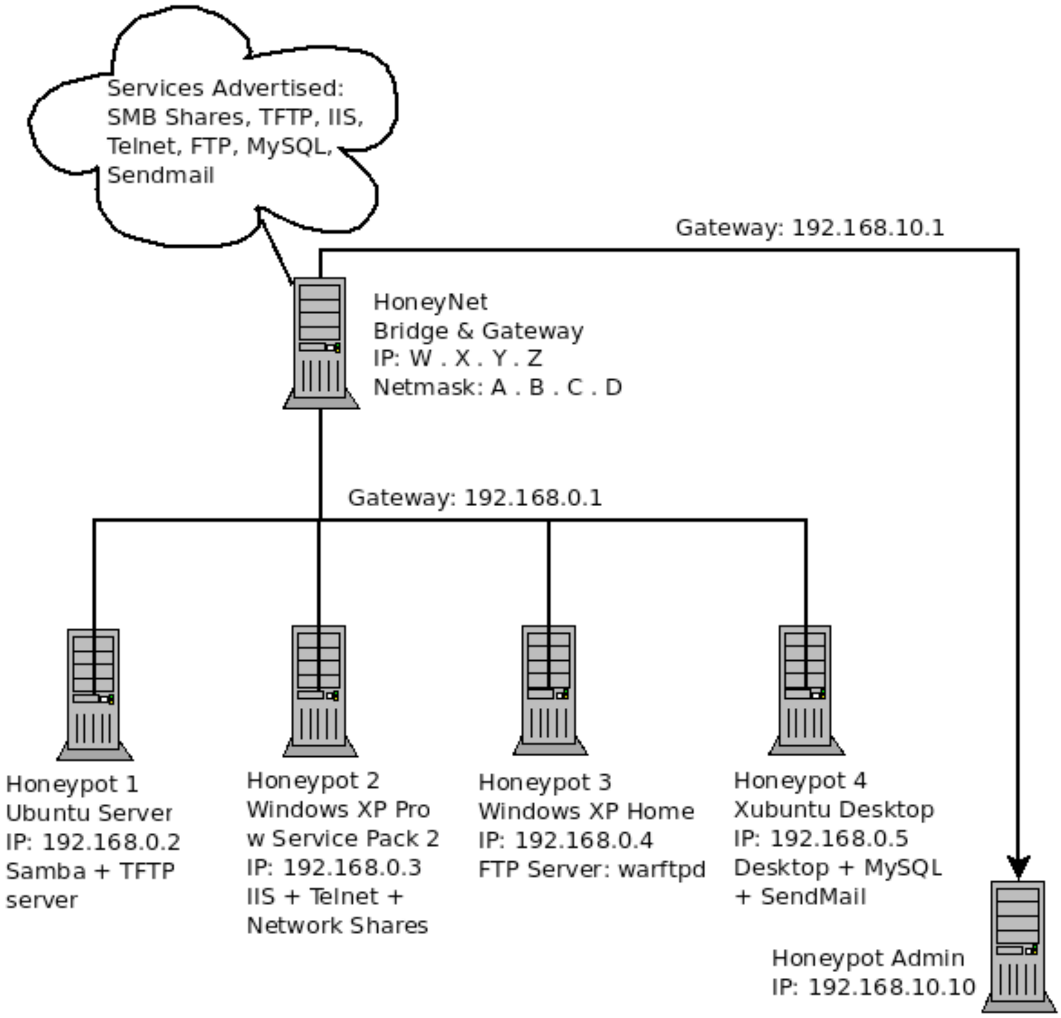

Figure 2: The structure & details of our proposed implementation

As it can be seen in Figure 2, the entire honeynet network is split up into two segments. The first contains the four computers acting as honeypot machines for the honeynet. The second one contains the machine responsible for administering the honeynet network. The honeypot machines in the first network segment have a variety of operating systems and network services running on them, emulating a small office setup.

This separation of the honeynet into two segments is fast becoming a standard feature of honeynet networks, according to the Honeynet project's discussion on 2nd and 3rd generation honeynets (The Honeynet Project 2006) and its purpose is to ensure the safety of the administration console from the rest of the honeynet network. This precaution is taken in the case of an attacker using it to compromise and take control of the entire honeynet network. Another reason for this precaution is to ensure the data going to and coming from the administration console are safe from eavesdropping.

Furthermore, the honeypot computers comprising the honeynet network
in question have been set up so as to not only not contain but also not gather any data that can be compromised.

4.1. The Honeynet Bridge

In the existing setup, the honeynet bridge is comprised of an Athlon XP Dual 3800+ workstation with a 200GB hard-drive capacity and 1GB RAM memory and three network interfaces, a WAN interface and two LAN interfaces, the first serving the honeynet network and the second serving the administration console (as seen on Figure 2).

The operating system it is running on is the PFSense unix firewall distribution (http://www.pfsense.org/) which is based on FreeBSD 7 but with the addition of the pf program which is the packet filtering mechanism used in OpenBSD (and ported to FreeBSD). The Pf packet filter is considered to be one of the most advanced open-source packet filters, with the ability to pre-process a packet prior to allowing it in the network. Its selection was based on that fact as well as the stability, customizability and small footprint of the Unix distribution which makes it most appropriate for use in a honeynet.

When dealing with security, and especially forensics, trust relationships are a crucial part of any system. While deploying PFSense as a honeynet bridge the premise that PFSense, as a system, is trusted, was accepted. However, it would be irresponsible to trust 3rd party plug-ins, thus the custom *rc.d* script method – which we have total control over – was chosen over an existing plug-in. This was largely due to concerns over the security and integrity issues of using a non-verified 3rd party plugin without proper testing, as opposed to the use of a simple and fully controllable tcpdump command.

This also boasts the ability of enabling administrators to send the capture files to a remote location or save them to locally-mounted remote shares – with slight adjustments to the *NETWORKING* file – hence creating a centralised database of captured packets.

With regards to the firewall responsible for the protection of the actual computers comprising the honeynet, is critical to create carefully planned rules both to avoid any legal issues, for example an attacker transforming the honeynet to a botnet, and to ensure the correct correspondence of service-to-honeypot.

Honeypot ID	Rule ID	Action	Proto	Source	Src. Port	Destination	Dest. Port	Gateway	State	Description
2	1	pass	TCP	*	*	192.168.10.3	80	Default	keep state	NAT IIS Server (WinXP Pro)
2	2	pass	TCP	*	*	192.168.10.3	23	Default	keep state	NAT Telnet Server (WinXP Pro)
3	3	pass	TCP/UDP	*	*	192.168.10.4	21	Default	keep state	NAT FTP Server (WinXP Home)
1	4	pass	TCP/UDP	*	*	192.168.10.2	137 - 139	Default	keep state	NAT SMB (Ubuntu 9.04)
4	5	pass	TCP/UDP	*	*	192.168.10.5	3306	Default	keep state	NAT MySQL (Xububtu)
4	6	pass	TCP	*	*	192.168.10.5	25	Default	keep state	NAT Sendmail (Xububtu)
*	7	Block	TCP/UDP	*	*	*	*	*	*	Implicit block

Figure 3: Firewall rules for services offered by Honeynet

However, there is a fine line that needs to be drawn between what we do want the attacker to be able to do and what should not allow the attacker to do. Figure 3 illustrates the firewall rules for our reference implementation, Those rules show that we *do* want an attacker to infiltrate our network and encourage him to be as malicious as humanly and technically possible. We also *do* want him to set up a botnet. But, we *should not* allow the botnet to be operational, so we need to contain it within out honeynet.

Hence honeypots should be unable to initiate outbound connections while enabling them to communicate with the attacker. Thus, stateful firewall rules will be enforced utilizing the Stateful Packet Inspection (SPI) technique. (Compex Systems Pte Ltd. 2010)

4.2. The Honeypot Machines

The computers acting as the honeypot machines in the honeynet were the following:

Honeypot 1:

- Hardware Setup: AMD Sempron 2200+, 512MB RAM, 120GB HDD
- OS: Ubuntu Server

- Version: 9.04
- Services running: smb, ftpd, tftpd

Honeypot 2:

- Hardware Setup: AMD Duron 1.10GHz, 512MB RAM, 10GB HDD
- OS: Windows XP Professional
- Version: Service Pack 2
- Services running: httpd (IIS), telnetd (terminal services server)

Honeypot 3:

- Hardware Setup: Intel Celeron 1.1GHz, 512MB RAM, 15GB HDD
- OS: Windows XP Home
- Version: -
- Services running: ftpd (warftpd)

Honeypot 4:

- Hardware Setup: Intel Pentium III, 250MB RAM, 20GB HDD
- OS: Xubuntu Desktop
- Version: 8.10
- Services running: mysqld, sendmail

As you can see from the above list, the machines acting as honeypot computers are both very low-end and relatively old machines, with the relatively recent ones acting as desktop workstations with a few services enabled, and the older ones acting as servers, as tends to be common practice on SME offices.

It is important to note, at this point, that the operating systems of the honeypot computers have not in any way been enhanced with data capturing facilities. The reason behind the use of what one might call "vanilla" configurations is for these systems to appear to be as benign as possible to someone who will at one point or another connect to, and ultimately exploit and gain full access to, them.

4.3. The packet capturing methodology

The PFSense system itself features a wide range of plug-ins, some of them designed to assist in capturing packets. However, in this system none of those were used – instead a simple addition was made to the "*NETWORKING*" script located at "*/etc/rc.d/NETWORKING*".

This file is by default empty – serving as a place-holder – and in order to accommodate our packet capturing needs it was edited as follows in Figure 4.

This simple one-liner at the end of the file in Figure 3 will run at boot time, capturing all packets traveling through the WAN interface, *em0*, and save them in the user's home directory.

```
#!/bin/sh
#
# $FreeBSD: src/etc/rc.d/NETWORKING,v 1.13.4.1 2008/01/28 07:58:31 dougb Exp $
#
# PROVIDE: NETWORKING NETWORK
# REQUIRE: netif netoptions routing network_ipv6 isdnd ppp
# REQUIRE: routed mrouted route6d mroute6d resolv
#     This is a dummy dependency, for services which require networking
#     to be operational before starting
#
#
echo "[CUSTOM] Starting tcpdump deamon…"
/usr/sbin/tcpdump -i em0 -w ~/em0_capture_`date '+%Y%m%d%H%M%S'`.cap &
```

Figure 4: The /src/etc/rc.d/NETWORKING script

In this instance the file would be saved in the */root* directory with the filename *em0_capture_20100619015950.cap*.

The network data packet capturing solution that was required in order to capture enough network packet information without going into extremes, we found, was the standard tcpdump utility. By itself, without using the “-vvv" option, the normal invocation of the tcpdump command allowed a sufficient degree of packet header and packet payload information. The full command used is shown in Figure 4, and as you can see the resulting log file stores the year, month, day, hour, minute and second of each dataset file in .cap format.

4.4. The Tar Pit

As it has been mentioned before, the network segment where the honeypot computers are located is set up to only allow entities from outside the private network to access the computers in the private network by means of the services each one has enabled. That, in turn, means that the outside entities can only interact with the individual computers hosting these services and cannot use those computers to gain access to other computers in the private network.

It also means, however, that the outside entities cannot use the services offered by the computers in the private network as a proxy to access services or computers outside the private network, thus essentially becoming boxed in in a LaBrea Tar Pit, as this particular network set up is known.

The concept of a LaBrea Tar Pit, or “sticky honeypot” as it is also known as, was introduced by Liston (2001) in response to the CodeRed worm, when security researchers needed to find some way of analyzing the worm’s properties, as Ruvalcaba (2009), explains in his paper on combining smart intrusion detection systems with LaBrea tarpits.

This particular network set up has been implemented as a safety feature to stop attackers from using the network as a staging point for further attacks and thus implicate the owners of the honeynet network in any malicious activities.

One of the added advantages of using the LaBrea Tar Pit model as it was used in this honeynet network is that while the attackers can request information (eg. webpages) they cannot maintain a constant 2-way transaction over any length of time, such as a full FTP session, as while something can enter the network it cannot leave the network.

5. Testing the Honeynet and analysis of data gathered

The honeynet network was tested on a 2Mbit DSL line and network data was gathered for 8 months between June 2009 and January 2010.

The presense of the honeynet network as a small office network was "advertised" on a network security website simply as a link marked "Private" in the main menu of the web site. Otherwise, the network was left untouched apart from weekly restarts of the NETWORKING service so as to harvest the network data captures.

The period of time was chosen specifically because it allows for seasonal trends and variations in the captured data resulting from the start and finish of the summer holiday period for most of the working and non-working populace and then leading up to, and including, the Christmas holiday period.
Figures 5a and 5b illustrate the fluctuations in the number of packets captured over the period of time the honeynet was running. In total, 19 million packets were captured.

The spikes and slumps in network traffic that can be more clearly seen in Figure 5b show the high and low periods of network activity observed in the honeynet. Specifically, given that by definition a honeynet only receives increased traffic from suspect parties, the spikes show periods of increased illegitimate network activity, such as automated and manual network attacks.

Initial analysis of the resulting datasets shows that the existence of the honeynet as an apparently insecure network of computers is "discovered" within the first two days of its operation, whereupon it begins to pick up traffic mostly aimed at the web server of the honeynet. The traffic then increases during the next few days, with regular port-scans and brute-force type password guessing attacks against most services, as well as attempts to make some use of these services.

Over a period of three to four weeks, specifically between the end of May and the end of June, network traffic dwindles to a trickle of exploratory requests for network services and picks up again and in quantities more or less equal to the previous spikes but with a higher ratio of password-guessing attacks against network services, and higher ratio of port-scans within a smaller range of ports. At the same time there were a number of attempts to upload content to the honeypot computer acting as the un-passworded FTP server.

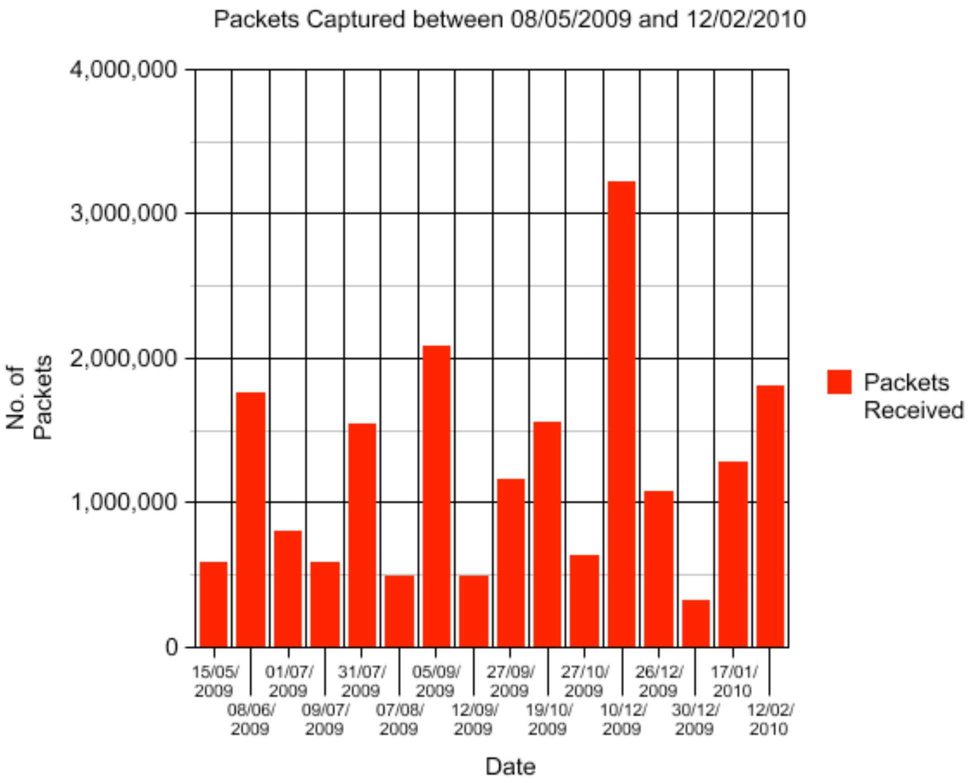

Figure 5a: Packets Captured between May 2009 and February 2010

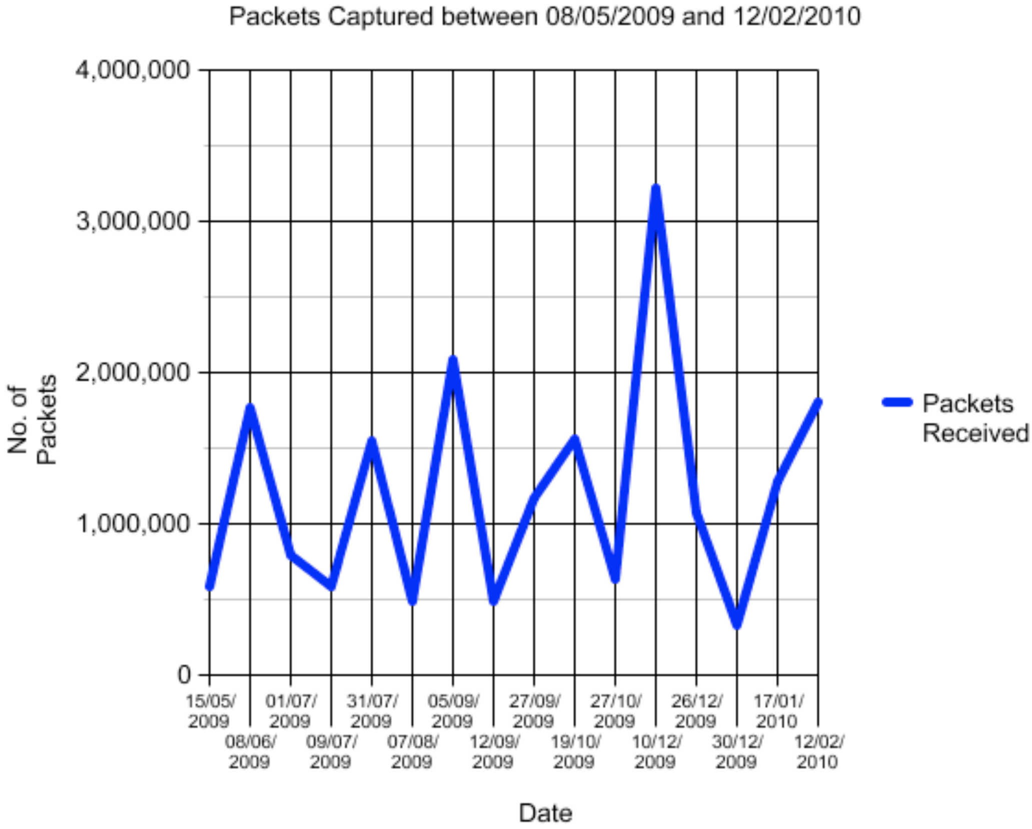

Figure 5b: Packets Captured between May 2009 and February 2010

Over a period of three to four weeks, specifically between the end of May and the end of June, network traffic dwindles to a trickle of exploratory requests for network services and picks up again and in quantities more or less equal to the previous spikes but with a higher ratio of password-guessing attacks against network services, and higher ratio of port-scans within a smaller range of ports. At the same time there were a number of attempts to upload content to the honeypot computer acting as the un-passworded FTP server.

The above trend continues, with little variation in the number and type of traffic, all the way through the rest of the summer and early autumn, however it is important to note that the datasets collected throughout this period of time are starting to show an increasing number of different IP addresses performing both attempted penetration and exploitation of services, and attempts at uploading content to particular services, namely our un-passworded FTP server. During the same period of time we are also beginning to observe highly targeted attempts to access and exploit the Microsoft Directory Services range of ports and the beginnings of attempted access to and exploitation of the MySQL database service.

The last two observations are especially predominant in the data captured in the period of time starting from the beginning of September and up to the end of December, by which point the two attacks dominate the network traffic captured.

Finally, in the remaining period of data collection running up to the end of January, a third and equally large-scale and highly targeted brute-force attack is observed, this time against the webserver. This ran concurrently with the previously dominant attacks against the Microsoft Directory Services and the MySQL database, but coming in from different IP addresses to the other attacks.

6. Conclusions & future work

Our reference implementation has shown that commodity and even legacy hardware can be used for the honeypots. Indeed, the whole honeynet environment, from routers and firewalls to the honeypot machines, can be constructed from off-the-shelf hardware and software, both cutting costs and making the honeynet environment more realistic.

Since the honeynet is an actual network comprising of servers and workstations it looks and behaves like a production network, therefore its attractiveness to attackers will be greater than a low-interaction honeypot and honeynet solution. The additional firewall rules and the application of a stateful-inspection firewall, as presented, will adequately contain all illicit activity inside the honeynet while allowing attackers the freedom they need to compromise the network.

Furthermore, the network capturing procedure is both simple and highly adjustable, with the potential for the resulting honeynet bridge to employ highly customised and scripted network packet capturing functionality through the use of a free, established and highly regarded network packet capturing program.

As a next step, a modified PFSense distribution focused on Honeynets could be developed to assist with data gathering and packet capturing. A PFSense plug-in could be developed to supplement the current plug-in, featuring extensive packet capturing capabilities while also guaranteeing forensic integrity of the data gathered. Virtualization technologies such as VMWare, and XEN have made a significant impact in how systems are being deployed. Thus, there is no reason why honeynets should make use of its merits as well.

Also, through the use of virtualisation technologies we can create honeynets-in-a-can with a honeynet bridge based on either the vanilla PFSense distribution or the customised distribution mentioned earlier, while the honeypots would be pre-configured to be deliberately vulnerable. In such a case, a single physical machine could incorporate an entire honeynet that would be able to run out-of-the-box.

References

ACPO, (2007), “Good Practice Guide for Computer Based Electronic Evidence V4.0”, The Association of Chief Police Officers (ACPO) / 7safe, (http://www.7safe.com/electronic-evidence), Last seen: 19/6/2010

Compex Systems Pte Ltd, (2010), "Stateful Packet Inspection Firewall", (http://www.nowire.se/produktblad/Compex/Compex_Firewall.pdf), Last seen: 19/6/2010

Cooke E., Jahanian F., McPherson D., (2005), “The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets”, Proceedings of the SRUTI ’05: Steps to Reducing Unwanted Traffic on the Internet Workshop, Cambridge, MA, pp. 39 - 44

Dornseif M., May S.A., (2004), “Modeling the costs and benefits of Honeynets”, June 2004, in Proceedings of the 3th Annual Workshop on Economics and Information Security (WEIS04), University of Minessota, US, (http://www-i4.informatik.rwth-aachen.de/lufg/publications/files/2004-weis-honeyeco.pdf), Last seen: 10/06/2010

Krasser S., Grizzard J.B., Owen H.L., (2005), “The Use of Honeynets to Increase Computer Network Security and User Awareness”, March 2005, Journal of Security Education, Volume 1, Issues 2 & 3, pp. 23 – 27

Kuwatly I., Sraj M., Masri Z.A., (2004), “A Dynamic Honeypot Design for Intrusion Detection”, in Proceedings of The IEEE/ACS International Conference on Pervasive Services, pp.95-104 (http://doi.ieeecomputersociety.org/10.1109/PERSER.2004.3), Last seen: 10/06/2010

Liston, T., (2001), “LaBrea: “Sticky” Honeypot & IDS”, (http://labrea.sourceforge.net/labrea-info.html), Last seen: 10/06/2010

Marshall A., (2008), “Digital Forensics: Digital Evidence in Criminal Investigations”, John Wiley & Sons Ltd., UK, ISBN: 978-0-470-51775-8

Manzano Y., Yasinsac Al., (2003), "Honeytraps, a Valuable Tool To Provide Effective Countermeasures for Crime Against Computer and Network Systems", July 2003, *Proceedings of the 7th World Multi-conference on Systemics, Cybernetics and Informatics* (SCI), (http://www.cs.fsu.edu/~yasinsac/Papers/MY02.pdf), Last seen: 10/06/2010

Nepenthes, (2009), “Nepenthes”, March 2009, (http://nepenthes.carnivore.it/), Last seen: 10/06/2010

O’Leary M., Azadegan S., Lakhani J., (2006), “Development of a Honeynet Laboratory: a Case Study”, June 2006, Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'06), Las Vegas, Nevada, US, pp. 401-406

Pang, R., Allman, M., Paxson, V. & Lee, J. (2006), “The devil and packet trace Anonymization”, SIGCOMM Comput. Commun. Rev. 36(1), 29-38. (http://doi.acm.org/10.1145/1111322.1111330), Last seen: 01/06/2010.

Paxson, V. (2004), `Strategies for sound internet measurement', IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, pp. 263 – 271, (http://doi.acm.org/10.1145/1028788.1028824), Last seen: 1/06/2010.

Portokalidis G., Bos H., (2007), “SweetBait: Zero-Hour Worm Detection and Containment Using Low-and-High-Interaction Honeypots”, April 2007, Computer Networks: The International Journal of Computer and Telecommunications Networking, Volume 51, Issue 5, pp. 1256-1274

Provos N., (2004), “A Virtual Honeypot Framework”, August 2004, The 13th USENIX Security Symposium, San Diego, CA, US, (http://www.citi.umich.edu/u/provos/papers/honeyd.pdf), Last seen: 10/06/2010

Ren W., Jin H., (2005), “Honeynet Based Distributed Adaptive Network Forensics and Active Real Time Investigations”, Proceedings of the 2005 ACM symposium on Applied computing, Santa Fe, New Mexico, SESSION: Computer-aided law and advanced technologies (CLAT), pp. 302-303

Riebach S., Rathgeb E.P., Toedtmann B., (2005), “Risk Assessment of Production Networks Using Honeynets – Some Practical Experience”, April 2005, ISPEC'05 - 1st Information Security Practice and Experience Conference, Singapore, pp. 1 – 12.

Ruvalcaba C., (2009), “Hybrid IDS – LaBrea Tarpit”, December 2009, SANS InfoSec Reading Room, SANS Institute, (http://www.sans.org/reading_room/whitepapers/casestudies/smart-ids-hybrid-labrea-tarpit_33254), Last seen: 10/06/2010

Sicker, D. C., Ohm, P. & Grunwald, D. (2007), `Legal issues surrounding monitoring during network research', IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement pp. 141 – 148, (http://doi.acm.org/10.1145/1298306.1298307), Last seen: 1/06/2010.

Spitzner L., (2003), “Honeypots: Definitions and Value of Honeypots”, May 2003, Honeypots:

Tracking Hackers, (http://www.tracking-hackers.com/papers/honeypots.html), Last seen: 17/09/08

The Honeynet Project, (2006), “Know Your Enemy: Honeynets”, May 2006, Honeynet Project, (http://www.honeynet.org/papers/honeynet/index.html), Last seen: 10/06/2010

The Honeynet Project, (2005), “Know Your Enemy: GenII Honeynets”, May 2005, Honeynet Project, (http://www.honeynet.org/papers/gen2/index.html), Last seen: 10/06/2010

The Honeynet Project, (2005), “Know Your Enemy: Honeywall CDROM Roo”, August 2005, (http://old.honeynet.org/papers/cdrom/roo/index.html), Last seen: 10/06/2010

Thonnard O., Dacier M., (2008), “A framework for attack patterns' discovery in honeynet data”, September 2008, Digital Investigator, Vol. 5:1, pp., 128-139

Zhuge J., Holz T., Han X., Guo J., Zou W., (2007), “Characterizing the irc-based botnet phenomenon”, December 2007, Peking University & University of Mannheim Technical Report, (http://www.honeynet.org.cn/downloads/publication/TR_IRC_Botnet.pdf), Last seen: 10/06/2010