

Regis University ePublications at Regis University

All Regis University Theses

Spring 2011

The Best Nix for a Combined Honeypot Sensor Server

Stephen M. Rodriguez
Regis University

Follow this and additional works at: <https://epublications.regis.edu/theses>



Part of the [Computer Sciences Commons](#)

Recommended Citation

Rodriguez, Stephen M., "The Best Nix for a Combined Honeypot Sensor Server" (2011). *All Regis University Theses*. 632.
<https://epublications.regis.edu/theses/632>

This Thesis - Open Access is brought to you for free and open access by ePublications at Regis University. It has been accepted for inclusion in All Regis University Theses by an authorized administrator of ePublications at Regis University. For more information, please contact epublications@regis.edu.

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

Use of the materials available in the Regis University Thesis Collection ("Collection") is limited and restricted to those users who agree to comply with the following terms of use. Regis University reserves the right to deny access to the Collection to any person who violates these terms of use or who seeks to or does alter, avoid or supersede the functional conditions, restrictions and limitations of the Collection.

The site may be used only for lawful purposes. The user is solely responsible for knowing and adhering to any and all applicable laws, rules, and regulations relating or pertaining to use of the Collection.

All content in this Collection is owned by and subject to the exclusive control of Regis University and the authors of the materials. It is available only for research purposes and may not be used in violation of copyright laws or for unlawful purposes. The materials may not be downloaded in whole or in part without permission of the copyright holder or as otherwise authorized in the "fair use" standards of the U.S. copyright laws and regulations.

THE BEST NIX FOR A COMBINED HONEYPOT SENSOR SERVER

A PROJECT

SUBMITTED ON 13 OF APRIL, 2011

TO THE DEPARTMENT OF INFORMATION TECHNOLOGY OF THE SCHOOL OF

COMPUTER & INFORMATION SCIENCES

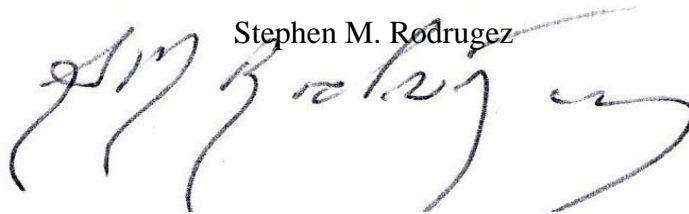
OF REGIS UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF MASTER OF SCIENCE IN

COMPUTER INFORMATION TECHNOLOGIES

BY

Stephen M. Rodrugez



APPROVALS

D. M. Libanish

Advisor Name, [Thesis/Project] Advisor

Douglas I. Hart

Douglas I. Hart

Shari A. Plantz-Masters

Shari Plantz-Masters

Project Paper Revision/Change History

Date	Project Status or Change
04/22/ 2008	Initial Project Meeting with ILB Project Lead (Jeff Brown)
04/25/ 2008	Decided on Solaris 10 for Honeypot sensor server on a Sun E-450
04/29/ 2008	Began hardware/ OS software configuration on Sun E-450
05/21/ 2008	Completed hardware, OS software, and network configuration of Sun E-450
06/04/ 2008	Submitted formal Idea document with the initial Annotated Bibliography
06/17/ 2008	Began installation of Honeypot sensor server software
07/12/ 2008	Submitted formal Project Statement of Work (SOW)
08/21/ 2008	Worked with sancp developer on compiling issue with Solaris 10
09/17/ 2008	Determined pads component would not compile on Solaris 10
10/23/ 2008	Determined needed tcl and tclx versions were not compatible on Solaris 10 Abandoned Sun Solaris 10 as workable solution for Honeypot sensor server
11/04/ 2008	Began installing Honeypot sensor software on HP running Red Hat Linux
11/21/ 2008	Submitted Thesis and Chapter 1 for Review
11/28/ 2008	Completed Honeypot sensor software install
12/03/ 2008	Revised Thesis statement based on shift in project scope
12/11/ 2008	Sucessfully tested Honeypot sensor server via remote Honeypot client software
1/15/ 2009	Began data collection/ compilation
6/5/2009	Began drafting paper
12/29/2009	Completed final draft
1/7/ 2010	Completed final edits
11/1/2010	Administrative Review and Return for edits
3/14/2011	Final Draft Submitted
4/13/2011	Format Corrections

Abstract

The paper will examine (through case-study) the usability of open source operating systems software for a combined Honeypot sensor server. The study will scrutinize the use of two Unix variants, Linux Red Hat and the Sun Solaris operating systems as candidates for deployment of a combined Honeypot sensor server. Appropriate unbiased metrics, such as extensibility, reliability, ease of install and use, will be employed as a likely criterion to evaluate the operating systems for the role of hosting Honeypot sensor server software.

Acknowledgements

It has been a privilege to attend an institution based in the Christian ethos of service to the community and the Lord. On many occasions at Regis University, I have seen the Spirit in action. I have seen instructors, students, and faculty give generously of their time, knowledge, and their kindness towards a stranger's success. As that stranger, I am grateful and honored to have partook of this blessing. In particular, I would like to thank Daniel Likarish, Jeffrey A. Brown, Todd Edmands, and Annette Argo without whose contributions, this project would have been a success.

I thank my family for their support over these many years of study. As a family, may we enjoy the benefits of this shared success. The goal has always been the freedom and opportunity that knowledge bestows. In closing, may we know God's love, light, and eternal peace...

Table of Contents

Abstract.....	6
Acknowledgements	7
List of Figures.....	9
List of Tables	10
Chapter One: Introduction and Project Background Summary	11
Problem Statement.....	12
Qualitative Case Study	14
Relevance of Project	15
Project Barriers.....	16
Proposal and Scope.....	16
Risks	16
Document Organization	17
Definition of Acronyms.....	18
Chapter Two: Research.....	20
Honeypot Overview	20
Honeypot Components	22
Heart of the Honeypot: The Sensor.....	22
The Choice of the Honeypot Sensor Server OS.....	24
Criteria for OS Analysis.....	25
Chapter Three: Configuration.....	28
Hardware Configuration.....	28
Software Configuration	36
Chapter Four: Wrapping it Up	40
Results	40
Summary.....	45
Recommendations.....	45
Bibliography	47

List of Figures

Figure 1: Logical network diagram of an... (Rodriguez, 2008).....12

Figure 2: Honeypot Data Lifecycle (Rodriguez, 2009)..... 21

Figure 3: Example Honeypot sensor network configurations (Rodriguez, 2008).....23

Figure 4: OS Version Information (Rodriguez, 2008).....24

Figure 5: Combined Honeypot Sensor Server OS Criterion (Rodriguez, 2009).....25

Figure 6: Disk Layout and Partitions (Rodriguez, 2009).....36

List of Tables

Table 1: Simplified OS Decision Matrix (Rodriguez, 2008).....	13
Table 2: Sun Enterprise[tm] 450 Server Hardware Spec...(Rodriguez, 2009).....	33
Table 3: HP ProLiant DL380 G5 Server series...(HP, 2009).....	36
Table 4: Sguil Software Components (Bianco, 2008).....	39
Table 5: Extensive OS Decision Matrix (Rodriguez, 2009).....	40

Chapter One: Introduction and Project Background Summary

This project encompassed the development, configuration, test, and verification of an Internet-facing Honeypot sensor server that can be used within a future Regis University Information Assurance (IA) lab or within the curriculum. Although, this project began with a grander scheme of examining the veracity of a completely functional Internet Facing Honeypot Network, the project shifted to determining the best operating system to host a combined Honeypot sensor server software suite, based on the technical difficulties encountered using an established Unix vendor as the “first-choice” operating system.

The development of the Internet-facing combined Honeypot sensor sever would reinforce concepts, methods, and techniques taught at different levels of IA coursework through education, awareness, and hands-on training. The Honeypot sensor server is only part of a larger Honeypot system, so the expectation is that this initiation project will be followed by future Regis SEAD students that will continue to build out the remaining Honeypot system. The choice of the operating system for the sensor sever is critical for the Honeypot system success; this paper will attempt to demonstrate the pros and cons of two flavors of popular server operating systems. Figure 1 is a logical network diagram of an Internet-facing Honeypot system (Rodriguez, 2008).

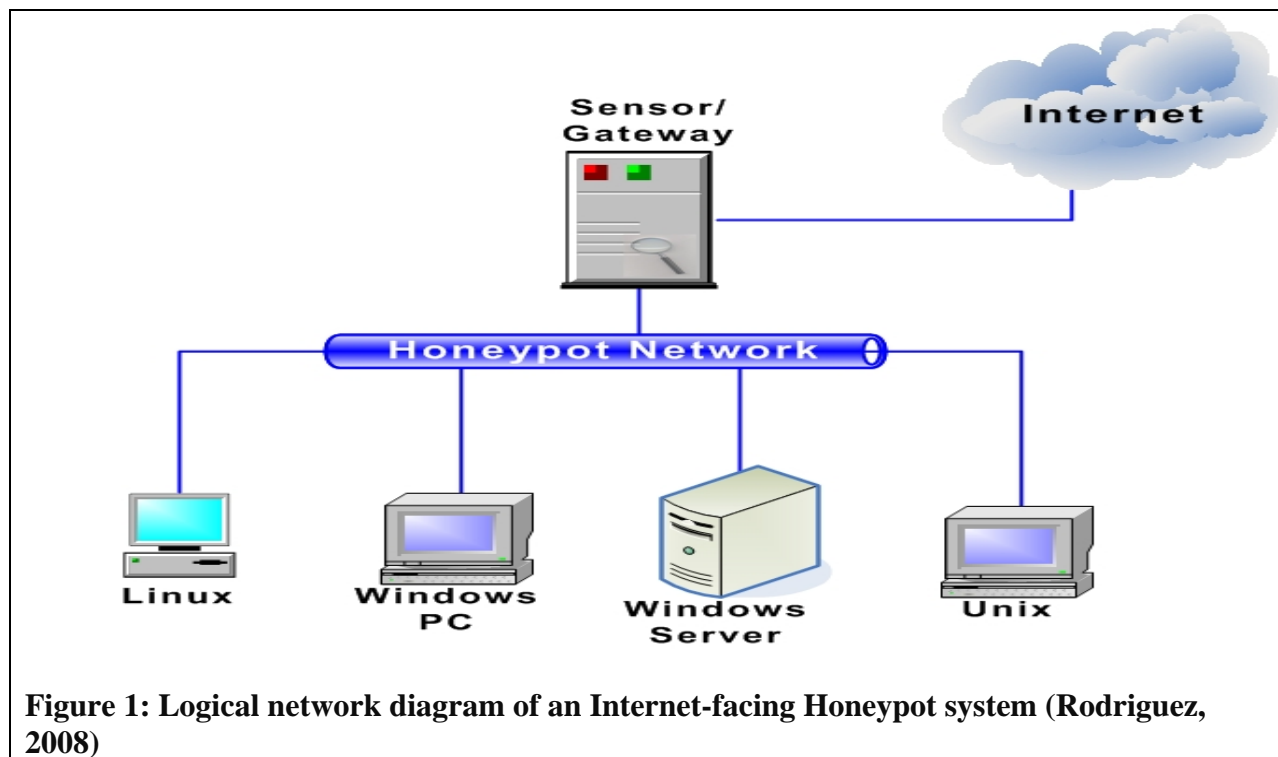


Figure 1: Logical network diagram of an Internet-facing Honeypot system (Rodriguez, 2008)

Problem Statement

The study will document and provide metrics to determine the usability of open source operating systems software for a combined Honeypot sensor server. It is purposed that the study will examine the use of two Unix variants, Red Hat Linux and the Sun Solaris operating systems as candidates for deployment of a combined Honeypot sensor server. Suitable unbiased metrics, such as extensibility, reliability, ease of install and use, will be examined as a likely criterion to evaluate the operating systems as a viable Honeypot sensor server candidate.

Potential Solutions

Due to the importance of the Honeypot sensor server to the Honeypot network, the choice of a suitable operating system is paramount to the success of the system. The Honeypot software chosen for this exercise, purported to support a myriad of Unix and Linux variant (Visscher, 2007). At the time of the project's commencement, the choice of suitable hardware was limited, which tended to limit the OS selection for the sensor server.

Sun Solaris and a variety of Linux distributions were considered as the OS of choice, for the combined Honeypot sensor sever (Brown, 2008). The OS determination was made, based on the hardware available, the reputation of a proven operating system, individual familiarity, and the access to a variety of support mechanisms (see Table 1).

Operating System (OS)	Hardware Support	Database Support	Software Support	Individual Experience
Solaris	YES (NATIVE)	YES (NATIVE)	YES	YES
Red Hat	NO	YES	YES	YES
Ubuntu	YES	YES	YES	NO
Gentoo	NO	UNKNOWN	UNKNOWN	NO

Table 1: Simplified OS Decision Matrix (Rodriguez, 2008)

Initially, Sun was selected, based hardware support, familiarity, native database support, and other qualifying criteria (see Figure 2. Simplified Decision Matix). Sun is a stalwart in the Unix industry having contributed heavily to the Unix computing environment, with solutions such as NFS, NIS, Java, etc.(Sun Microsystems, Inc., 2008). Since Regis had recently received a donation of Sun equipment and its operating system was being offered as a free download, it seemed a reasonable candidate to host the combined Honeypot sensor server software. Though Sun Solaris was initially decided on as the OS of choice for the Honeypot sensor server, it was abandoned after months of failed attempts to compile all the needed software components for the system.

Red Hat is one of the premier providers for enterprise class Linux distributions. Since its first release in 1994, Red Hat has continued to grow and win numerous awards (Red Hat, 2008). Ultimately, Red Hat Linux was used and successfully supported this project.

Qualitative Case Study

This project will present a qualitative case study of the configuration of a HoneyPot sensor server, employing a research methodology that utilizes an evidence-based analysis. The research methodology chapter, demonstrates the types of evidence observed and measurables identified. The study will attempt to remove the subjective (e.g., individual experience with a product) and place value on empirical research, though it is acknowledged that a totally objective analysis is not feasible when considering factors such as ease of install, use, supportability, etc.

Relevance of Project

The criticality of information assurance (IA) in an organization continues to gain greater importance in the enterprise. With this, comes the ever-increasing challenge for educational institutions to output quality graduates able to meet the task. New threats and risks to an enterprise's IA posture are continually being developed and exploited. One educational solution to this ongoing threat to enterprise information assurance is to provide students with a real-world environment, where threats are constantly evaluated and risk mitigation actively explored. As has been empirically demonstrated, a student's learning is enhanced through hands-on experience, experimentation, and in-depth labs (Fisher, 2004). The intent of this project is to deliver an operational Internet-facing, combined Honeypot sensor server that will provide a learning environment where students can exercise new and existing IA skills. The student will accomplish this, by analyzing current threats and develop techniques to minimize risks to the organization.

The selection, configuration, and delivery of the Honeypot sensor server is the foundation for this project's current and ongoing relevance, key to this success is the choice of operating system for the sensor. An operating system/ network operating system (OS/ NOS) is the basis for the security triad of confidentiality, integrity, and availability (Dulaney, 2009). In the absence of a secure OS, the Honeypot's worth would be diminished, if not completely negated. Besides affording an interface to the hardware and a command execution environment, today's secure network operating systems also provide authentication, accounting, and availability (Bovet & Cesati, 2006):

Project Barriers

As with any low-funded and understaffed educational endeavor, there are many barriers that accompany such a project. The first challenge was acquiring suitable software and hardware to support the project. Because of budgetary constraints, open source and freely available software was utilized. The hardware requirements are dictated by software and yet, still need to be robust enough to handle the traffic that will traverse a fully operational, Internet-facing Honeypot sensor server. Thus, the hardware available at the time of project commencement, had a direct correlation on the software selected.

Proposal and Scope

There are four phases of the project work in this case study:

Phase I: Hardware selection, installation, configuration, and verification.

Phase II: Operating System selection, installation, configuration, and verification.

Phase III: Honeypot sensor server software selection, installation, configuration, and verification.

Phase IV: Final Honeypot sensor server verification and testing.

Risks

In the end, this project strives to increase security within the organization/ enterprise. As Lance Spitzner states, “security is all about reducing risk” (Honeypots : tracking hackers, 2003, p. 321). This project presents some technical risks to Regis University by placing the Honeypot system within the organization’s infrastructure and exposing it to the Internet. Spitzner identifies three risk factors with Honeypot systems (Honeypots : tracking hackers, 2003):

1. Level of Interaction: hackers are given full access to the system and can possibly compromise system beyond configuration restraints.

2. Complexity: the Honeypot system is comprised of many complex elements (firewalls, rulesets, access control lists, etc.) and requires competent system/ network administration controls.
3. Network exposure: the Honeypot system is Internet accessible and can provide a means for hackers to access an organization's internal network.

Document Organization

The remainder of document will address the following topics:

- Research: data and information collection/ dissemination concerning this project.
- Configuration: setup and configuration of the hardware and software in support of this project.
- Results/ Recommendations: finding and final determinations of this project.
- Summary: project synopsis and review.

Terms

Definition of Acronyms

Acronym	
CERT	Not an acronym. CERT term is owned by Carnegie Mellon University, and is part of the Software Engineering Institute
CPU	Central Processing Unit
DoD	Department of Defense
GUI	Graphical User Interface
HPC	High Performance Computing
IA	Information Assurance
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IPS	Intrusion Prevention System
LAN	Local Area Network
MAC	Media Access Control
NIC	Network Interface Card
NIDS	Network Intrusion Detection System
NOS	Network Operating System
OS	Operating System/s
RAID	Redundant Array Of Independent Disks
ROI	Return On Investment
RPM	Red Hat Package Manager
SANS	SysAdmin, Audit, Network, Security Institute
SCSI	Small Computer System Interface
SQL	Structured Query Language
TCO	Total Cost of Ownership
US-CERT	United States Computer Emergency Readiness Team
WAN	Wide Area Network
WWW	World Wide Web

Definition of Terms

Term	Definition
Authentication	Authentication requires users to prove their identity
Extensibility	
Framework	Frameworks provide structure and guidelines
Information Assurance	Measures that protect and defend information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. These measures include providing for restoration of information systems by incorporating protection, detection, and reaction capabilities. (National Information Assurance Glossary)
Information Security (IS)	Protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction. Information security is concerned with confidentiality, integrity, and availability. (CITE)
Measurement	In this study, the data collections are both internal and external to the environment
Model	Models are conceptual, and do not provide any direction or guidelines
Protocol	A protocol is an agreed upon format for transmitting data between two devices
Reliability	A data collection strategy in qualitative study that requires stability and the creation of creatable procedures which is accomplished with a formal case study
Security	Security is described through the accomplishment of some basic security properties, namely confidentiality, integrity, and availability of information. (Kotzanikolaou and Douligeris)
Security Architecture	The design artifacts that describe how the security controls are positioned and how they relate to the overall IT architecture. These controls serve the purpose to maintain the system's quality attributes, among them confidentiality, integrity, and availability. (CITE, 2008)
Standards	Written definition or rule approved for compliance by consensus or by authoritative groups

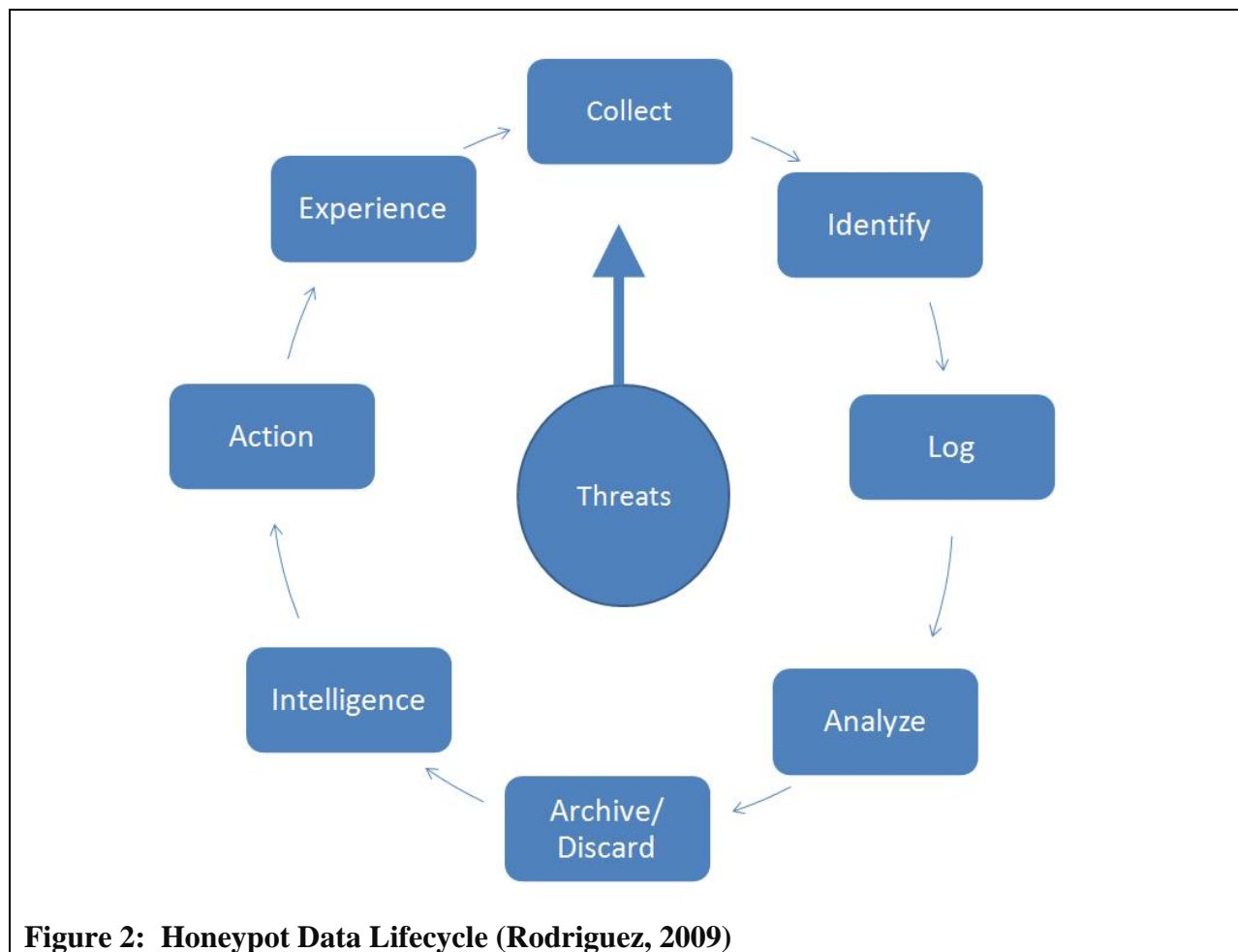
Chapter Two: Research

Honeypot Overview

Much can be found in print and online, detailing research and proving the concepts of Honeypot systems. "A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource" (Spitzner, 2003). In essence, a Honeypot network is an intentionally designed, security-flawed network, composed of a variety of vulnerable sub-systems and computers. Its objective is to delay, divert, and draw attackers to a central point by the use of subterfuge. It can be used to as a means of legal entrapment against would-be hackers (Dulaney, 2009). Oftentimes, the Honeypot contains false data (e.g., spreadsheets, employee lists, accounting information, etc.) that is left within the vulnerable system.

At this point, it makes sense to define the difference between the terms Honeynet and Honeypot. A Honeynet is a high-interaction implementation of a Honeypot (Spitzner, 2002). However referenced, both are considered a NIDS (Network Intrusion Detection System) and play a role as part of the network security paradigm.

A Honeypot can directly support the SecSDLC (The Security Systems Development Life Cycle) and as a result, an organization's security policy. Furthermore, a Honeypot accomplishes this by supplying a means for investigation, analysis, design, implementation, maintenance, and change (Whitman & Mattord, 2005). Similarly, figure 4 depicts the Honeypot data lifecycle, from the threats entry into the Honeypot, to the final desired effect (knowledge).



Honeypot Advantages

The Honeypot's ultimate benefit is that of knowledge and experience. Unless the organization is building products off the information gathered, it generally does not provide any production value to an organization. Honeypots, when effectively utilized (Whitman & Mattord, 2005):

- Can obtain useful information on the methods of attackers, hackers, and intruders.
- Can be used to identify network risks and vulnerabilities.
- Can be used to identify current methods and techniques employed by hackers.
- Can be used to aid in incident response, forensics, and legal prosecution of computer/network espionage.

Honeypot Disadvantages

Honeypots do have their limitations. First, Honeypots are not reliable countermeasures for enterprise security, meaning, they are not an IPS (Intrusion Prevention System). In addition, they can have the effect of taunting attackers and result in an increase in the severity of attacks on the enterprise. Lastly, they expose a part of the organization's network to intruders and may compromise an organization's security strategy, or provide inadvertent information on a further means of attack. For example, if the sensor or other network device is compromised, it may provide a backdoor to an organization's intranet (Whitman & Mattord, 2005).

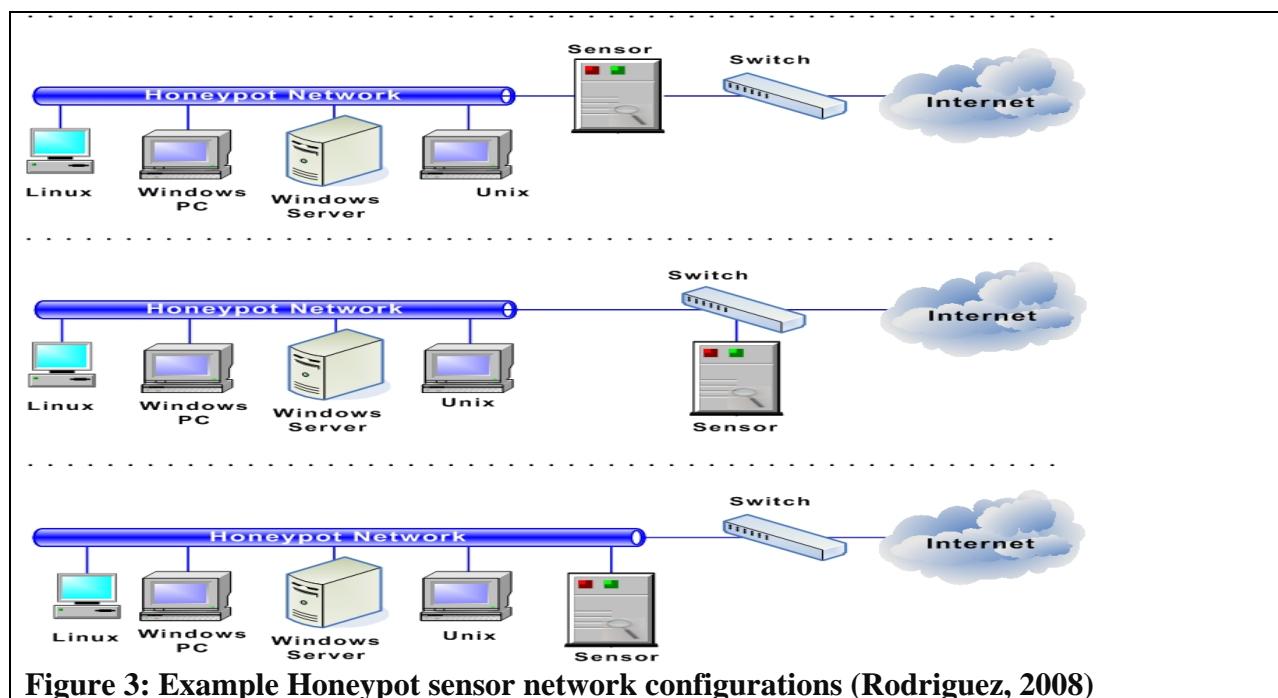
Honeypot Components

As can be derived from Figure 3, the Honeypot is composed of distinct components or modules. The system can be simplified into three sections, the Honeypot sensor, the server, and the client (Visscher, 2007). The the sensor monitors and collects network packets. In addition, the sensor identifies data based on rules, also known as signature-based monitoring (Dulaney, 2009). The server has the ability to log, store (archive), or discard the data. The server contains the database component of the Honeypot system and facilitates the auditing capability of the product. Lastly, the client provides a means for the human element, where the Honeypot user/administrator can turn raw data into actionable information and ultimately gain knowledge.

Heart of the Honeypot: The Sensor

As previously conveyed, the focus of this writing is towards the configuration of the sensor server (consolidation of first two sections of the Honeypot system). The sensor is the heart of the Honeypot network. Its job is to capture and monitor packets traversing the Honeypot network. The Honeypot sensor can be setup in various network configurations: directly in-band to the traffic path (as a type of pass-through router), out-of-band on a network switch (port

monitoring), or as another node on the Honeypot network (packet sniffer). See figure 4 for the respective configuration examples.



Due to its mission, the sensor (or combination sensor server) should be robust enough to support this type of role. The sensor software dictates the minimum hardware requirements, as does most software. Many of today's server class computers, can be adequately configured for this purpose. Common to most Honeypot servers, is the need for (Hoepers & Steding-Jessen, 2006):

- A highly robust multi-tasking, multi-processor server.
- RAID disk arrays to store the large amounts of data collected, availability, and provide for the necessary through-put of high traffic scenarios.
- Depending on configuration, at least one high-speed network interface (see Figure 4).
- Suitable hardware to support the requirement for a database server.

There are many suitable Honeypot software suites/ packages. Many are freely available and based on open source software license agreements. They can be compiled to run on a variety of

hardware platforms and operating systems. A sampling of Honeypot software, can be found at The Honeynet Project: <http://project.honeynet.org/project>.

The Choice of the Honeypot Sensor Server OS

The focus of this paper's research is centered on the choice of OS, for a combined Honeypot sensor server. The selection of the OS is driven by the need to combine analogous and disparate software modules that can be compiled, to become the Honeypot system. The OS needs to support an open environment, in the sense that from developers to end-users, the OS should not impede the success of the project.

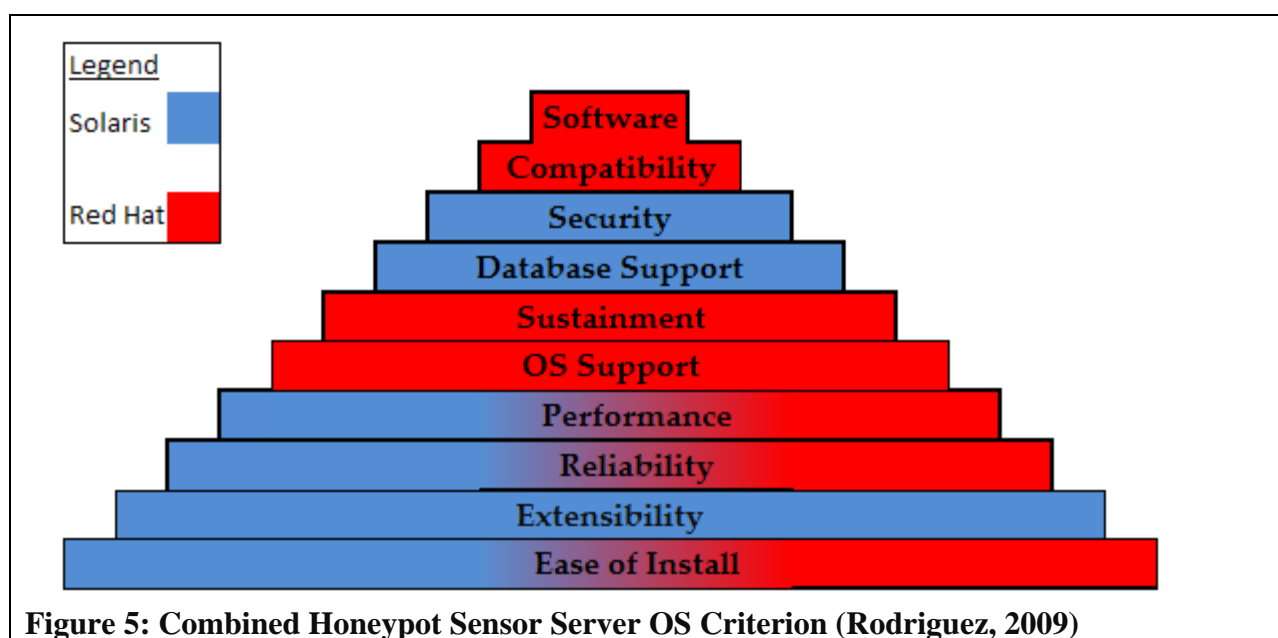
Two established operating systems were considered for the implementation of the combined Honeypot Sensor Server: Sun Solaris 10 and Red Hat Enterprise Linux 4.1.2. Both operating systems are leaders in their industry and good Unix and Linux representatives. Both are innovative and have contributed to the continuing strength and success of the Unix and Linux operating systems. The intent of this study is not to determine what OS is necessarily better, but to identify the OS that was better suited for this specific project. The parameters of this study were limited to one particular Honeypot software suite and which OS was more readily suited to support the success of this project. See figure 5 for the command outputs that display OS version information (for Red Hat Linux and Sun's OS, respectively).

<pre>[srodri506@centaur ~]\$ uname -a Linux centaur.vlab.us 2.6.18-92.1.22.el5 #1 SMP Tue Dec 16 12:03:43 EST 2008 i686 i686 i386 GNU/Linux</pre>
<pre>[srodri506@centaur ~]\$ cat /proc/version Linux version 2.6.18-92.1.22.el5 (mockbuild@builder16.centos.org) (gcc version 4.1.2 20071124 (Red Hat 4.1.2-42)) #1 SMP Tue Dec 16 12:03:43 EST 2008</pre>
<pre>root:sensor# uname -a SunOS sensor 5.10 Generic_137111-08 sun4u sparc SUNW,Ultra-4</pre>

Figure 4: OS Version Information (Rodriguez, 2008)

Criteria for OS Analysis

As the study proceeds in the following chapters, a set of OS characteristics will be presented that were found to be pertinent to the development and configuration of the combined Honeypot sensor server. Figure 6, is a pictorial representation of the OS characteristics that were found valuable in the successful deployment in the Honeypot project. The elements of the pyramid (Figure 6) are categorized into: ease of install, extensibility, reliability, and performance.



Ease of install is a relative determination of the ease in which the OS was installed, based on a familiarity with current and past OS installations over a 20 year period. Both Solaris 10 and Red Hat 4 possessed equally intuitive install GUIs that made installation of the OS straightforward and problem-free. Ease of install is important to the Honeygot, due to the fact that Honeygot systems are meant to be attacked, compromised, and rebuilt.

Extensibility is the degree in which a system can be modified to adopt to future requirements, while maximizing an organization's ROI (Cornish, et. al., 2003). Since 2000, Red Hat has not officially supported Sun's Sparc CPU, which equates to a decrease in platform support (Shankland, 2000). Sun's has continued to ink important hardware partnerships, IBM longtime cooperation with Sun being one of many examples (Vaughan-Nichols, 2007). With Oracle's acquisition of Sun, Solaris is poised to expand its market share. Sun ultimately has the advantage in this area, by supporting more types of architectures and platforms (Babcock, 2008). Extensibility is important to the longevity and resilience of the Honeygot project.

Reliability is a measure of the product's dependability, to ensure system uptime. Both Solaris 10 and Red Hat are mature operating systems with successful track records of enterprise support. Reliability supports the Honeygot's availability. The Honeygot's effectiveness is directly related to its uptime.

Performance is the efficiency, resource effectiveness, and comparative speed of the operating system to its competitors. There is very little definitive research giving one operating system an overall performance advantage over the other. For example, looking at Sun's own research comparing their latest ZFS file system to other common Linux file systems, results in the all-too-common, "depending on the application" commentary (Sun, 2007). Also, many of the comparisons seemed to be testing systems that were just too different in hardware configurations,

thus nullifying the results (Principled Technologies, 2007). Performance is important to the Honeypot system, because of the resources (CPU, memory, and I/O) required to handle the load of a network-based attack. For example, a DOS (denial of service) attack is based on bombarding a networked computing device with data packets, to the point the system becomes unusable.

OS Support is the ability and the cost related to software maintenance. Though Sun argues that their overall TCO is less than Red Hat, it is undeniable that an open OS affords more avenues to support than the traditional software model. The findings of this project were that Red Hat was more prevalent and maintainable. With the advent of OpenSolaris, this result may change in the future with Sun's participation of the open source movement. Open source software benefits from a community of free developers and testers. Redhat has better leveraged this model through the use of its beta OS'. The supportability of the OS is directly related to the longevity of the project.

Sustainment is the ongoing ability to support a project in a cost-effective manner. Where supportability addresses the technical issues of maintaining the project, sustainability tackles the logistics of what it would take to maintain the project from a personnel/ resource point of view. For example, Solaris system administrators wear very specialized suits, as Sun continues to bring their own brand of uniqueness and innovation to the table (e.g., RBAC, Zones, Dtrace, etc.). Red Hat's innovations (e.g., RPM) have been quickly adopted by the Linux community, thus avoiding the "members only" mentality or the requirement for specialized training.

The database houses all the data, needed for the operation of the Honeypot. The data is mined, analyzed, and signatures built on traffic patterns, so database support is a principal consideration. Database support is the ability to maintain and support the required database for

the Honeypot system. Sun is owned by Oracle, a leading database provider. And, with Sun's acquisition of MySQL in 2008, Sun now has the ability to bundle MySQL with their OS as a low-cost alternative to Oracle suite of database products (MySQL.com, 2008).

Security is the ability for the system/ OS to provide confidentiality, integrity, and availability (Dulaney, 2009). Being able to secure the Honeypot server is critical for this project to provide a viable environment for study, testing, and learning. Back in 1995, Sun released its first version of what it termed "Trusted OS" (Brunette, 2006). Sun has continued to improve the security within the Solaris platform with a myriad of tools and concepts on access, auditing, accounting, allocation controls. (Sun, 2009). For example, through zoning, Sun allows easily built and secured virtual environments. Again, Linux is not far behind, but Sun has a slight lead in this area (Babcock, 2008).

Software Compatibility is the OS' ability to support the required Honeypot software. The majority of the Honeypot software packages were easily configured with Red Hat; whereas, trying to compile the needed packages and versions on Sun was difficult, and some cases not possible at the time of this study (April 2008- October 2008). Examples of install and configuration can be found in the appendices.

The result of this study (based on the selected Honeypot software) was conclusive. Red Hat provided a more suitable OS for this project. As figure 6 demonstrates, both operating systems have their advantages, but on the more critical issues of software compatibility, software/ OS support, and ongoing sustainability, Red Hat is clearly the OS of choice. Lastly, Sun has only a slight advantage over Red Hat in the areas of security and database support.

Chapter Three: Configuration

Hardware Configuration

Since the project focused on the configuration of an Internet-facing combined Honeypot sensor sever, the type of hardware chosen would have to fulfill an important role. The hardware would require significant disk space to host the possibly massive log collection and database growth. The system would need sufficient resources to handle the data bursts that could accompany a malicious network attack. Of course the system would need multi-processing/multi-tasking capability to address all the software requirements.

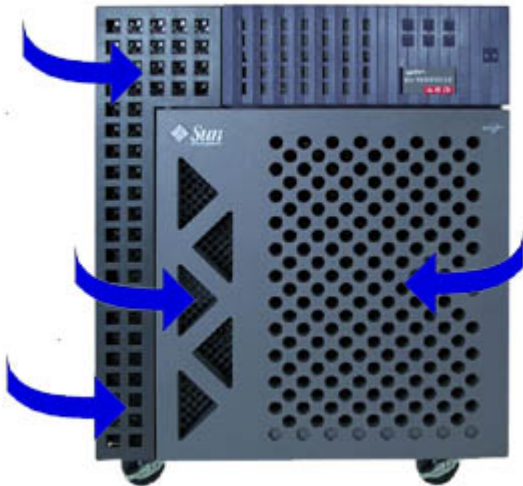
Sun Enterprise[tm] 450 Server

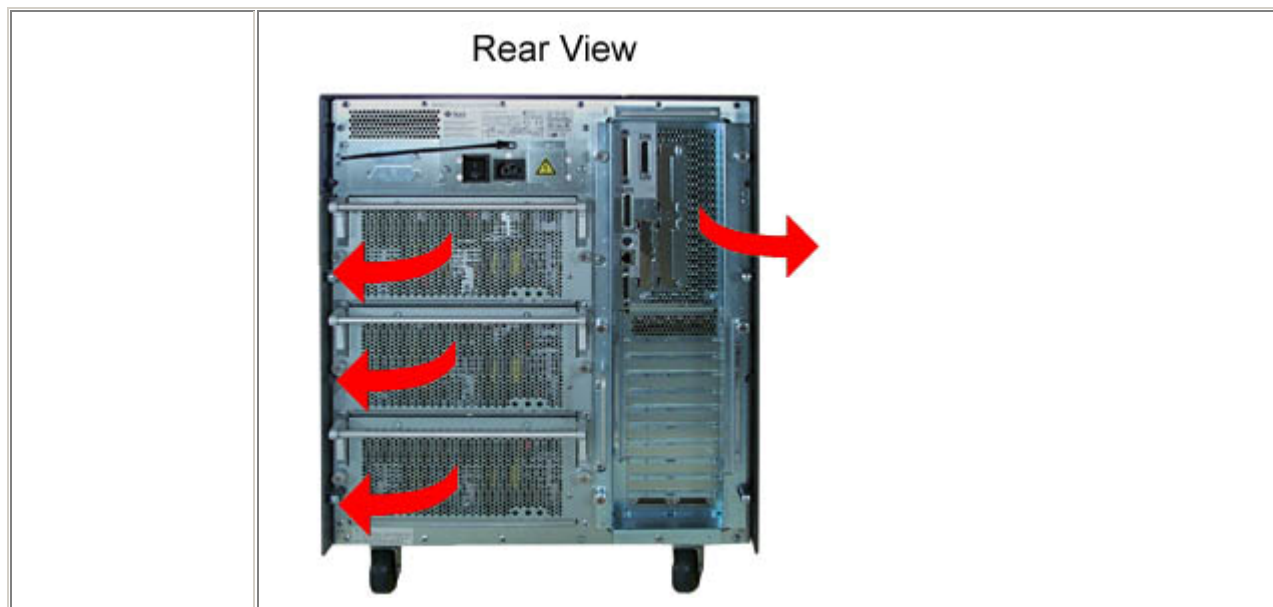
The Sun Enterprise[tm] 450 Server debuted in September of 1997. Although outdated by today' standards, the server met the minimum requirements and was a good candidate to handle the load. The sever also had adequate storage capabilities and potential for expansion (via built-in SCSI adapters). More importantly the hardware was readily available for this project.

Processor	
Number	From one to four processor modules
Architecture	250-, 300-, 400- or 480-MHz UltraSPARC[tm]-II modules with onboard E-cache
Cache memory	16-KB I-cache, 16-KB D-cache per processor 1-MB external cache per processor with 250-MHz CPU 2-MB external cache per processor with 300-MHz CPU 4-MB external cache per processor with 400-MHz CPU
Datapath	Two independent, buffered 144-bit UPA buses; 128 bits data, 16 bits ECC; two processors per bus UPA operates at 100-MHz with 300-MHz or 400-MHz processors
Main Memory	
Capacities	16 DIMM module slots; four banks of four slots

	<p>Accepts 32-, 64-, 128-MB or 256-MB DIMMs</p> <p>128 MB to 4 GB total memory capacity</p>
Memory type	144-pin 5V 60-ns memory modules
Datapath	<p>576 bits wide; 512 bits data, 64 bits ECC</p> <p>Up to 1.78-GB/sec throughput</p>
Standard Interfaces	
Ethernet	One ethernet/fast ethernet (10BASE-T/100BASE-T) twisted-pair standard connector (RJ-45) or one MII for external transceiver connection, autoselect port
Keyboard and mouse	One standard keyboard/mouse port (mini DIN-8)
Parallel	One Centronics compatible, bidirectional, EPP port (DB25)
PCI	<p>Ten slots compliant with PCI specification version 2.1:</p> <p>Three slot operating at 33- or 66-MHz, 32- or 64-bit data bus width, 3.3 volt</p> <p>Four slots operating at 33-MHz, 32- or 64-bit data bus width, 5 volt</p> <p>Three slots operating at 33-MHz, 32-bit data bus width, 3.3 volt</p>
SCSI	<p>One 20MB/sec, 68-pin, Fast/Wide SCSI-2</p> <p>One, three, or five 40-MB/sec, UltraSCSI-3 buses for internal disks</p>
Serial	Two RS-232D/RS423 serial ports (DB25 , requires a Y-type splitter cable)
Internal Mass Storage	
Disks	<p>Up to twenty 4.2-GB, 9.1-GB, 18.2-GB, or 36.4-GB (3.5- x 1-in.) hot-swap UltraSCSI-3 drives</p> <p>Disk bays: Four, twelve, or twenty hot-swap disk bays</p> <p>Disk controllers: One, three, or five 40-MB/sec UltraSCSI-3 channels; maximum four drives per channel</p>
CD-ROM	SunCD[tm] 12x or 32x 644-MB SCSI CD-ROM (standard)
Floppy	1.44-MB 3.5-in. floppy drive (standard)
Tape	One bay available for optional 5.25- x 1.6-in. SCSI tape drive; 8-mm or 4-mm DDS-3, or SLR

Power Supplies	
Type	One, two, or three modular, N+1 redundant, hot swap, universal input (two supplies standard)
Output	1210W maximum, 605W maximum each supply 1120W maximum, 560W maximum each supply (before October 1997)
Power bus	Common, load-sharing
Environment	
AC Input	100 - 240 VAC, 47 - 63 Hz, 13.8 A(max)
Input Power	1664 W
Heat Output	5680 BTU/hr
Temperature ¹	Operating: 5° C to 35° C (41° F to 95° F) Nonoperating: -20° C to 60° C (-4° F to 140° F)
Humidity	Operating: 20% to 80% relative humidity, noncondensing Nonoperating: 5% to 93% relative humidity, noncondensing
Altitude	Operating: 3000 m (10,000 ft.) Nonoperating: 12,000 m (40,000 ft.)
Acoustic noise	Operating: 6.9 bels Idling: 6.3 bels
Vibration	Operating: 0.2G peak, 5 - 500 Hz, 3 perpendicular axes Nonoperating: 1G peak, 5 - 500 Hz, 3 perpendicular axes
Shock	Operating: 4G peak, 11 milliseconds half-sine pulse Nonoperating: 30G peak, 11 milliseconds half-sine pulse
Number of cords	1
¹ The front and rear doors of the cabinet must be 63% open for adequate airflow.	
Regulations	
Safety	UL 1950 and CB-scheme EN60950 with Nordic Deviations, CUL C22.2 No. 950, TUV EN60950
RFI/EMI	FCC Class B, Industry Canada Class B, EN55022/CISPR22 Class B, VCCI Class B
Immunity	EN50082-1/IEC1000-4-2, IEC1000-4-3, IEC1000-4-4, IEC1000-4-5
Harmonics	EN61000-3-2/IEC1000-3-2

X-ray	DHHS 21 Subchapter J, PTB German X-ray Decree
Dimensions and Weights	
Height	58.1 cm (22.87 in.)
Width	44.8 cm (17.64 in.)
Depth	69.6 cm (27.40 in.)
Weight	94.0 kg (205 lb.)
Power Cord	2.5 m (8.2 ft.)
Clearance and Service Access	
Front ¹	36 in. (91.44 cm.)
Rear ¹	36 in. (91.44 cm.)
Right ¹	36 in. (91.44 cm.)
Left ¹	36 in. (91.44 cm.)
Top ¹	36 in. (91.44 cm.)
Airflow	<p style="text-align: center;">Front View</p>  <p>The image shows a front view of a Siemens X-ray unit. It is a dark grey, rectangular device with a control panel at the top and a large perforated grille on the front. Three blue arrows indicate the airflow pattern: one arrow points into the top left corner, another points into the left side panel, and a third points out of the right side panel. The Siemens logo is visible on the front panel.</p>



¹ These specifications refer to a system that is fully extended from the rack for service. When in normal operation, there are no side clearance requirements for the server as the air flow is from the front to rear. However, make sure that any front or back cabinet doors are 63% open to allow adequate airflow. This can be accomplished by removing the doors, or ensuring that the doors have a perforated pattern that provides a 63% open area.

Rack Mounting


The Sun Enterprise 450 can be mounted in a standard 19-in. rack. The optional rackmounting kit consists of a depth-adjustable, slide-mounted shelf and retaining bracket.

Table 2: Sun Enterprise[tm] 450 Server Hardware Specifications (Rodriguez, 2009)

HP ProLiant DL380 G5 Server

Following a number of unsuccessful attempts to install the Honeypot software on the Sun system (not hardware related), an HP ProLiant DL380 G5 Server was used. One of the major advantages of the HP ProLiant line of servers is the embedded hardware RAID controller, which minimized configuration requirements. In comparison, the Sun Enterprise[tm] 450 Server required the configuration of a slower software-based RAID controller.

Processor & Memory	
Processor Type	Intel® Xeon® 5400 series Intel® Xeon® 5300 series Intel® Xeon® 5200 series
Processor	<p>Quad-Core Processors</p> <p>Intel® Xeon® processor X5470 (3.33 GHz, 1333MHz, 120W)</p> <p>Intel® Xeon® processor X5460 (3.16 GHz, 1333MHz, 120W)</p> <p>Intel® Xeon® processor X5450 (3.00 GHz, 1333MHz, 120W)</p> <p>Intel® Xeon® processor E5450 (3.00 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor E5440 (2.83 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor E5430 (2.66 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor L5430 (2.66 GHz, 1333MHz, 50W)</p> <p>Intel® Xeon® processor E5420 (2.50 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor L5420 (2.50 GHz, 1333MHz, 50W)</p> <p>Intel® Xeon® processor E5410 (2.33 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor L5410 (2.33 GHz, 1333MHz, 50W)</p> <p>Intel® Xeon® processor E5405 (2.00 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor X5365 (3.00 GHz, 1333MHz, 120W)</p> <p>Intel® Xeon® processor X5355 (2.66 GHz, 1333MHz, 120W)</p> <p>Intel® Xeon® processor E5345 (2.33 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor L5335 (2.00 GHz, 1333MHz, 50W)</p> <p>Intel® Xeon® processor E5335 (2.00 GHz, 1333MHz, 80W)</p> <p>Intel® Xeon® processor E5320 (1.86 GHz, 1066MHz, 80W)</p> <p>Intel® Xeon® processor L5320 (1.86 GHz, 1066MHz, 50W)</p> <p>Intel® Xeon® processor E5310 (1.60 GHz, 1066MHz, 80W)</p> <p>Dual-Core Processors</p> <p>Intel® Xeon® processor X5270 (3.50 GHz, 1333MHz, 80W)</p>

	Intel® Xeon® processor X5260 (3.33 GHz, 1333MHz, 80W) Intel® Xeon® processor L5240 (3.00 GHz, 1333MHz, 40W) Intel® Xeon® processor E5205 (1.86 GHz, 1066MHz, 65W)
Processor Cores	Dual and Quad
Cache memory	Up to 12MB L2 cache (2 x 6MB)
Sockets	2
Max front side bus	1333MHz
Memory Type	PC2-5300 DDR2 FB DIMMs
Standard memory	2GB (2GB base models; 4GB performance models)
Max Memory	64GB
Memory protection	Advanced ECC; Mirrored Memory; Online Spare
Storage	
Storage type	Hot plug 2.5-inch SAS Hot plug 2.5-inch SATA
Max Drive Bays	Up to 8: SFF Hot plug to support Serial-attached SCSI (SAS) and Serial ATA (SATA) drives
Storage controller	Performance Models: HP Smart Array P400/512MB BBWC Controller (RAID 0/1/1+0/5/6) High Efficiency and Base Models: HP Smart Array P400/256MB Controller (RAID 0/1/1+0/5) Entry Models: HP Smart Array E200/64MB Controller (RAID 0/1/1+0)
Expansion Slots	4 total slots
Deployment	
Form factor	Rack
	
Rack height	2U
Networking	Two (2) Embedded NC373i Multifunction Gigabit Network Adapters with TCP/IP Offload Engine
Remote management	Integrated Lights-Out 2
Power supply type	Standard on performance models, optional on entry and base

	models
System fans	Hot plug fully redundant
Warranty - year(s) (parts/labor/onsite)	3/3/3

Table 3: HP ProLiant DL380 G5 Server series – specifications and warranty (HP, 2009)

As previously stated, the system required a fair amount of disk space and built-in reliability. The operating system disk was mirrored, utilizing a RAID 1 configuration. The data was striped across multiple disks, utilizing a RAID 5 configuration for redundancy. Figure 10 provides a pictorial representation of the disk layout and partition requirements.

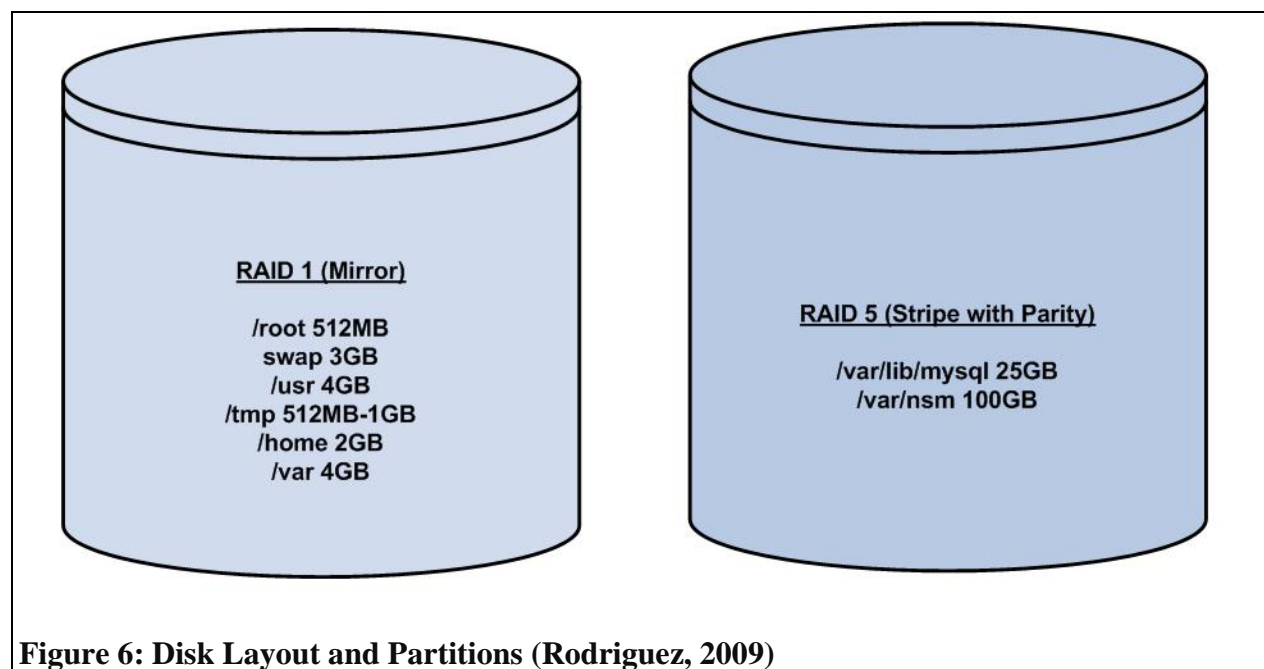


Figure 6: Disk Layout and Partitions (Rodriguez, 2009)

Software Configuration

Sguil (pronounced sgweel) was developed by network security analysts for network security analysts (Visscher, 2007). Sguil is a suite of modular applications that utilizes some of the best open source software available to comprise a network collection and monitoring security system. It is agile enough to incorporate new and better applications as they are developed and can be deployed on a number of different hardware and software platforms. Sguil provides the

functionality of an IDS, along with a myriad of data collection, and real-time monitoring tools. The Squil database allows for simple to complex SQL queries that can be completed in the shortest amount of time. All these tools and abilities are consolidated within the Squil GUI, which is the command center of the Honeypot system. In addition, Sguil provides an environment to develop, test, and analyze security tools and methodologies.

Both the Sun and Red Hat Sguil installs used the *Sguil on RedHat HOWTO* as an installation guide (Bianco, 2008). Though this particular website focuses on Red Hat, the processes and software components needed to install and configure Sguil are the same, regardless of the platform OS. The basic install steps were:

- Identify the needed software package/s.
- Download the source code
- Compile and install the software package/s
- Test and verify install

A list of the needed software components for Sguil can be found below:

Software	Version	Location	Download Location	Notes
MySQL	5.x	Server	http://dev.mysql.com/downloads/mysql	4.1.x versions also work
Libpcap	0.9.7	Sensor	http://www.tcpdump.org/	
Libnet	1.0.2a	Sensor	http://www.packetfactory.net/libnet/	Neither newer nor older versions may be used.
Snort	2.6.1.5 (or newer)	Sensor	http://www.snort.org/dl/	You will also need to register for a free snort.org account in order to download IDS rules

SANCP	1.6.1d	Sensor	http://www.metre.net/sancp.html	1.6.2 versions don't seem to work correctly for Sguil yet, but the developer is working on this.
Barnyard	0.2.0	Sensor	http://www.snort.org/dl/barnyard/	
PADS	1.2	Sensor	http://demo.sguil.net/downloads/pads-1.2-sguil-mods.tar.gz	You must use the version with the built-in Sguil modifications. The standard PADS version will not work.
P0f	2.0.8	Server	http://lcamtuf.coredump.cx/p0f.shtml	
Tcpflow	0.21	Server	http://www.circlemud.org/~jelson/software/tcpflow/	
Tcltls	1.5.0	Server, Client	http://tls.sourceforge.net/	Provides for an encrypted data channel between the sguil server and the analyst consoles or sensors
Mysqtlcl	3.03	Server	http://www.xdobry.de/mysqtlcl/	
Tcllib	1.9	Server	http://tcllib.sourceforge.net/	
Sguil	0.7.0	Server, Sensor, Client	http://www.sguil.net/	Note: 0.7.0 is currently in test release, so you'll need to fetch the CVS version .
Sguil startup scripts	0.7.0	Server, Sensor	http://instantnsm.sourceforge.net/	I've put together a set of prepackaged startup scripts for the

				various components. These files come with the InstantNSM distribution.
--	--	--	--	--

Table 4: Sguil Software Components (Bianco, 2008)

Due to Sun's prevalence, many of the required software components were found online, pre-compiled and ready for install (Christensen, 2009). Though, the Solaris 10 install of Sguil also required many software prerequisites that are not listed in Table 4. For example, TclX is a base component needed for Sguil and not part of the Solaris 10 full install. In the end it was a TCL version mismatch that forced the move to another OS. Since, TCL/ TclX are Sguil foundation modules, it became evident that the TCL mismatch might force a recompile/ reinstall of all the Sguil components. In addition, there were packages that would not compile on Solaris 10, the PADS application being one example (Meissner, 2008). Due to the unknown variables, lack of support from Sun/ Solaris community, and project time constraints (4 months were exhausted on the attempted Solaris Sguil install), Solaris 10 was abandoned (See Appendix A: Project Communications).

There are a slew of resources documenting the success of running Sguil on the Red Hat operating system. The benefit of being the more established Linux variant, clearly had a positive effect on the support available via news, forums, and user groups. Furthermore, the RPM package install utility was found to be much easier to navigate than Sun's "package add" utility for loading the additional software on the system. The Sguil software modules compiled and installed with much less user intervention (e.g., modifications to config/ make files) than on the comparable Solaris installation. As observed during this project, the hardware, software, and

application configurations were found to be more effective and time efficient using the Red Hat Operating System.

Chapter Four: Wrapping it Up

Results

This document studied the most adaptable Unix-like operating system for use as a combined Honeypot sensor server. From the plethora of operating systems considered, both Sun Solaris and Redhat Linux were selected for the study. In the end, Redhat Linux was better suited to accomplish the task of running a combined Honeypot sensor server. Previously in this paper, the evaluated OS criterion was represented pictorially (reference Figure 5: Combined Honeypot Sensor Server OS Criterion). Table 5 provides a more detailed examination of the OS comparison results.

		Operating Systems	
		Sun Solaris	Redhat Linux
Evaluated Characteristics	Ease of Install		X
	Extensibility	X	
	Reliability	X	
	Performance		X
	OS Support		X
	Application Sustainment		X
	Database Support	X	
	Security	X	
	Software Compatibility		X
	Hardware Flexibility		X
	Market Prevalence		X
	Requirement for Additional Training	X	

Table 5: Extensive OS Decision Matrix (Rodriguez, 2009)

Like Figure 5, the Extensive OS Decision Matrix (Table 5), illustrates the superior attributes of Redhat Linux over Sun Solaris. Table 5, was developed based on the need for a conclusive determination of OS characteristics, regardless of how slight the advantage of one OS over the other. In some cases the conclusion was based on real-world experience, survey, and perspective; not necessarily extensive empirical study. The reason for this is based on the fact that some characteristics are entirely relative to experience (e.g., ease of install). The following will review the main points that gave the prevailing OS the edge over the other and provide justification for the conclusion.

Ease of install is important to the Honeypot, due to the fact that Honeypot systems are designed to be attacked, compromised, and rebuilt. Though both OS were similarly easy to install, Redhat provided a slightly more user friendly interface based on its more Windows like interface. Again, both install interfaces were straight-forward, but from the perspective of a novice user, Redhat had a very slight edge.

Extensibility is important to the flexibility of the Honeypot project, since it will provide the foundation of where/ and what the Honeypot software can be installed. Development is directly related to overall platforms fielded. With Oracles commitment to maintain Suns hardware development/ deployment strategy, Sun has the opportunity to maintain its overall lead in CPU types that will support Sun Solaris (Finkle, 2009).

Reliability corresponds to the Honeypot 's availability or the uptime of the environment. The Honeypot's effectiveness will be serverly disabled, if the system is not deemed reliable. Solaris has a slight edge by the means of having the more mature operating system and experience in the enterprise with clustering, security, and less agile software releases. Although agile software

development is a benefit within some applications, the Honeypost system would benefit from Sun's more structured model of delivery.

Performance the speed and efficiency in which Honeypot system can do its job, measured in terms of hardware resource use: CPU, memory, and I/O. Though this study did not complete a side by side performance comparison, based on industry research, Linux has clearly dominated the HPC (High Performance Computing) market in recent years (Meuer, Strohmaier, Dongarra, & Simon, 2009).

The supportability of the OS is directly related to the longevity of the project. Technically, the project needs the best possible support model for its continued existence. Based on experience gained from this study, Linux is superior in this category, especially since the majority of Honeypot software was initially developed with the Linux operating system in mind.

Sustainment is the logistics to maintain the project from a personnel and economic perspective, differing from the technical aspects of supportability. Linux continues to lead in overall deployments and its growth looks to dwarf future deployments of Sun's current operating system model. This continues to lead to an increase in the number of people who know and can sustain Linux. Linux administrators will be more easily to find and train, than their Sun counterparts.

The Honeypot system is highly reliant on the collection of large amounts of data. Due to the necessity for the large collection of data, database integrity is crucial to the system. With Oracle's acquisition of Sun and with Sun acquiring, Sun seems poised to benefit from these newfound database partnerships (Finkle, 2009). Sun has a decisive database advantage over its competitors.

Confidentiality, integrity, and availability is core to the security of an operating system (Dulaney, 2009). Sun has a long tradition of providing a secure operating system environment. At least for the short-term, Sun maintains a slight security advantage over Linux, based on experience and overall OS maturity.

All things considered, it boils down to whether all the software components that compose the Honeypot system will work together. The myriad of software applications/ components that compose the Honeypot softwares suite were quite easily and successfully installed on Redhat. After numerous attempts, many of components, specific versions, would not compile on the Sun platform.

Hardware flexibility underlies the ability to operate the Honeypot system on a wide range of hardware platforms. While Sun might have an advantage in extensibility or the variance of CPUs that support Solaris, Redhat is openly supported by more Vendors. Redhat can be installed on just about any off-the-shelf computer. Redhat has a tremendous advantage over Solaris in the realm of hardware flexibility.

Market prevalence is a consideration, as the investment in the Honeypot system is significant. As previously stated, time, money, and other resources are required to maintain a system and the system should be designed with a healthy lifecycle at the forefront. Sun's once strong foothold in the government/ DOD sphere that once gave them an advantage has eroded as Redhat is now authorized in the U.S. government computing space (Beekman & Abhyankar, 2005). Redhat has continued to grow in number of commercial installs, as well (Kerner, 2009). Redhat continues to extend into many markets the use to be reserved for the big Unix giants (Sun, IBM, SGI, etc.).

Training requirements for any new system can delay its successful deployment, as teams are trained to maintain the system. Training and support through the Solaris community was once a model for others to follow. Many of those forums (e.g., Sun BigAdmin, SEToolkit, docs.sun.com, etc.) are now retired and all learning locked down to the paying community. As well, with the latest versions of Solaris, Sun has taken a strategy of foraging a new direction, separate from traditional Unix and Linux distributions. This strategy unfortunately requires specialized skills that results in a smaller pool of engineers, developers, and system administrators that are able to support the systems. A few examples:

- NFS file names are different (e.g., /etc/dfs/dfstab versus the traditional /etc/exports)
- Startup/ RC (run control) scripts are different (e.g., /etc/init.d has been antiquated in Solaris)
- Services management is different (svcs/ svcamdin versus /etc/config files)
- Network services (e.g., /etc/inetd.conf has been antiquated in Solaris)

Summary

From the results of this study, Redhat was the better product for this project. Clearly the benefits of having a massive pool of software contributors/ developers have given the open source OS community an edge over the traditional OS manufacturers. Specific evidence is documented in appendices, showing multiple Honeybot component application developers conceded that their applications were not being actively tested or developed for current Solaris' releases. The concessions from the legacy OS vendors (like Sun) to the open source movement are evident in the fact that every major OS manufacturer has jumped onto the open source bandwagon either by commitment or action. Visit Sun's open source project at <http://opensolaris.org>.

Recommendations

Recommendations for this project include:

- The primary use of Redhat or similar Linux distribution for a combined Honeybot sensor server
- The continued testing of new OS releases as they become available, to verify the system is current to the market
- The leveraging of virtual computing and its many advantages (i.e., hardware footprint, savings in HVAC, recovery time, system rebuild/ recovery time, etc.)
- Membership in one of the many international or national Honeybot groups/ alliances to further the organization's experience, knowledge, and contribution to ongoing security computing and networking efforts
- An active Honeybot lab, committed to collecting ongoing real-time data and developing counter-measures/ solutions to real world threats and vulnerabilities

- Separation of the Honeypot Server and Sensor to distinct/ individual computing environments (virtual or physical)

Bibliography

- Babcock, C. (2008). *Sun Shines In Solaris 10, Linux Comparison*. Retrieved February 16, 2009 from
<http://www.informationweek.com/news/software/linux/showArticle.jhtml?articleID=205207395>
- Beekman, M., & Abhyankar, S. (2005). *Red Hat and the Federal Enterprise Architecture*. Retrieved June 28, 2009 from http://www.redhat.com/f/pdf/gov/WHP0005US_FEA.pdf
- Bianco, D. J. (2008). *Sguil on RedHat HOWTO*. Retrieved May 2, 2009 from http://nsmwiki.org/Sguil_on_RedHat_HOWTO
- Bovet, D. P., & Cesati, M. (2006). *Understanding the Linux Kernel [3rd Ed.]*. Sebastopol, CA: O'Reilly Media.
- Brown, J. , A. (2008, 4 24). Personal Communication.
- Brunette, G. (2006). *There and Back Again – A Solaris Security Tale*. Retrieved March 2, 2009 from <http://nvd.nist.gov/scap/conf/2006/nist-secauto-solsec-v1.4.pdf>
- Chapple, M., Shinder, D. L., & Porter, S. (2003). *TICSA TruSecure™ ICSCA Certified Security Associate Exam TU0-001*. Indianapolis, IN: Que Certification.
- Christensen, S. (2009). *Sunfreeware.com: Freeware for Solaris*. Retrieved May 5, 2009 from <http://sunfreeware.com/>
- Ciampa, M. (2005). *Security+ Guide to Network Security Fundamentals (2nd Edition)*. Boston, MA: Thomson Course Technology.
- Cornish, R., Moore, T., Pavoni, D., and Rockenbach, E. (2003). *Analyzing Requirements and Defining .Net Solution Architectures Exam Cram™ 2(Exam 70-300)*. San Diego, CA: Pearson Education.

- Dulaney, E. (2009). *Comptia security+ study guide (4th Edition)*. New York: Sybex.
- Finkle, J. (2009). *Q+A-What are Larry Ellison's plans for Sun Micro?* Retrieved August 21, 2009 from <http://www.reuters.com/article/rbssTechMediaTelecomNews/idUSN0740285120090507>
- Fisher, B. (2004). *Seeing, Hearing, and Touching: Putting It All Together*. Retrieved October 30, 2008 from <http://www.cs.ubc.ca/~fisher/brfisher.work.html>
- Hewlett-Packard Development Company, L.P. (2009). HP ProLiant DL380 G5 Server series - Specifications and Warranty. Retrieved March 28, 2009 from <http://h10010.www1.hp.com/wwpc/us/en/sm/WF06a/15351-15351-3328412-241644-241475-1121516.html>
- Hoepers, C. & Steding-Jessen, K. (2006). *Distributed Honeypots Project: How It's Being Useful for CERT.br*. Retrieved January 26, 2008 from www.cert.org/archive/pdf/CERTbr-Honeypots-public.pdf
- Kerner, S. M. (2009). *Red Hat to Grow Linux Biz Via Partners*. Retrieved September 19, 2009 from <http://www.serverwatch.com/news/article.php/3835366/Red-Hat--to-Grow-Linux-Biz-Via-Partners>
- Meissner, D. (2008). *HELP with pads-1.2-sguil-mods on SPARC: msg#00025*. Retrieved September 12, 2008 from <http://osdir.com/ml/security.sguil.general/2008-09/msg00025.html>
- Meuer, H., Strohmaier, E., Dongarra, J., & Simon, H. (2009). *TOP 10 Sites for June 2009*. Retrieved July 17, 2009 from <http://www.top500.org/lists/2009/06>
- MySQL.com (2008). *Sun to Acquire MySQL*. Retrieved February 28, 2009 from <http://www.mysql.com/news-and-events/sun-to-acquire-mysql.html>

Principled Technologies, Inc. (2007). *WebBench performance on a Red Hat Enterprise Linux 5 Intel processor-based system and a Sun Solaris 10 AMD processor-based system.*

Retrieved February 25, 2009 from

<http://www.developers.net/intelisdshowcase/view/2559>

Red Hat (2008). *About Red Hat.* Retrieved October 26, 2008 from

<http://www.redhat.com/about/companyprofile/history/>

Shankland, S. (2000). *Red Hat drops Sparc support with new Linux version.* Retrieved February

12, 2009 from <http://news.cnet.com/2100-1001-249226.html>

Spitzner, Lance. (2002). *Honeypots : Tracking Hackers.* San Diego, CA: Pearson Education.

Spitzner, Lance. (2003). *Honeypots: Definitions and Value of Honeypots.* Retrieved March 16,

2009 from <http://www.tracking-hackers.com/papers/honeypots.html>

Sun Microsystems, Inc. (2007). *White Paper: SOLARIS™ ZFS AND RED HAT ENTERPRISE*

LINUX EXT3 FILE SYSTEM PERFORMANCE. Retrieved February 22, 2009 from

http://www.sun.com/software/whitepapers/solaris10/zfs_linux.pdf

Sun Microsystems, Inc. (2009). *About Sun.* Retrieved October 25, 2008 from

<http://www.sun.com/aboutsun/company/history.jsp>

Sun Microsystems, Inc. (2009). Sun Enterprise[tm] 450 Server: Hardware Specifications.

Retrieved March 27, 2009 from

http://sunsolve.sun.com/handbook_pub/validateUser.do?target=Systems/E450/spec

Vaughan-Nichols, S. (2007). *IBM, Sun Partner to Bring Solaris to IBM.* Retrieved January 15,

2009 from [http://www.eweek.com/c/a/IT-Infrastructure/IBM-Sun-Partner-to-Bring-](http://www.eweek.com/c/a/IT-Infrastructure/IBM-Sun-Partner-to-Bring-Solaris-to-IBM-Servers/)

[Solaris-to-IBM-Servers/](http://www.eweek.com/c/a/IT-Infrastructure/IBM-Sun-Partner-to-Bring-Solaris-to-IBM-Servers/)

Visscher, B. (2007). *Sguil: The Analyst Console for Network Security Monitoring*. Retrieved March 15, 2009 from <http://sguil.sourceforge.net/>

Whitman, M. E., & Mattord, H. J. (2005). *Principles of Information Security (2nd Edition)*. Boston, MA: Thomson Course Technology.