Securing Financially Sensitive Environments with OpenBSD

Nicholas C. P. Humphrey

Technical Report
RHUL–MA–2008–05
15 January 2008

Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX, England
http://www.ma.rhul.ac.uk/tech

Royal Holloway, University of London
MSc Information Security

Securing Financially Sensitive Environments with OpenBSD

Nicholas C. P. Humphrey

Supervisor: Prof. Peter Wild

Abstract

This thesis investigates the use of a free, open source UNIX-based operating system in providing security features to a financially sensitive business function such as a treasury.

We start by examining some of the main security features (such as the pf firewall and systrace policies) which are included with the operating system, how they work and how such features can be used within a financial environment. We then examine possible problems with each feature and the introduction of such a feature into the business environment. We also explore some of the criticism that OpenBSD has received and additional features which could be useful to business.

We then look at some examples of statutory and regulatory requirements, and how OpenBSD's features may be mapped to address such requirements. As part of this we examine how open source software in general can be utilised and some of the advantages and disadvantages of it against similar commercial offerings.

We then see a case study based on a real-world treasury, and some of the serious security concerns which are faced by security officers responsible for such departments. We explore how OpenBSD can be applied within an infrastructure to provide key security services and address some of the specific concerns raised in the treasury security assessment.

Finally, we provide conclusions and suggestions for future work.

Contents

1 Executive Summary **8**

I Introduction **10**

2 Introduction **11**
2.1 Project Overview 11
2.2 What is OpenBSD? 12
2.3 What is a Financially Sensitive Environment? 12
2.4 Project Objectives, Structure and Methodology 13
2.5 Document Production 14

II OpenBSD: Technologies and Features **15**

3 Overview **16**

4 OpenBSD History, Licensing and Source Access **17**
4.1 History 17
4.2 Licensing 18
4.3 Accessing the OpenBSD Source Code 18

5 The pf Firewall **20**
5.1 What is pf? 20
5.2 Using pf in an FSE security context 21
5.2.1 Ubiquitous Technology? 21
5.2.2 Auditing Connections 21
5.2.3 Data Flow Enforcement 22
5.2.4 Firewall Administration 22
5.2.5 High Availability 22
5.3 Problems with pf 23
5.4 pf - an example in practice 23
5.4.1 System Services 23
5.4.2 Target machine pf rule set 24

5.4.3 Results with pf enabled . . . 24
5.4.4 Results with pf disabled . . . 26

6 OpenSSH **28**
6.1 What is OpenSSH? . . . 28
6.2 Using OpenSSH in an FSE security context . . . 28
6.3 Problems with OpenSSH . . . 29

7 Systrace System Policies **30**
7.1 What is systrace? . . . 30
7.2 Using systrace in an FSE security context . . . 31
7.3 Problems with systrace . . . 32
7.4 Systrace - an example in practice . . . 32
7.4.1 A Simple Payments Processing Program . . . 32
7.4.2 Generating a Systrace Policy . . . 33
7.4.3 Modifying the Systrace Policy . . . 34

8 Memory Protection Features **36**
8.1 Features within OpenBSD . . . 36
8.1.1 W^X . . . 36
8.1.2 .rodata Segment . . . 36
8.1.3 Randomized malloc() and mmap() . . . 36
8.1.4 strlcpy() and strlcat() . . . 37
8.2 Using memory protection in an FSE security context . . . 37
8.3 Problems with memory protection features . . . 37

9 Auditing **39**
9.1 Auditing within OpenBSD . . . 39
9.1.1 syslog . . . 39
9.2 What additional auditing features are available? . . . 39
9.2.1 Output to Physical Log Printer . . . 39
9.2.2 Logging to Write-Once, Read-Many Media . . . 40
9.3 Using auditing in an FSE security context . . . 40
9.4 Problems with auditing . . . 40
9.4.1 Physical Logs Under Attack . . . 40
9.4.2 Dangers of Optical Media . . . 41
9.4.3 Generation of Audit Trails . . . 41

10 Other Relevant Features **43**
10.1 Redundant Array of Inexpensive Disks (RAID) . . . 43
10.2 Network Time Protocol (NTP) . . . 44

11 Hardening OpenBSD **45**
11.1 sshd Configuration 45
11.1.1 SSH v1 and Man-in-the-Middle 45
11.2 Disabling Unnecessary Services 46
11.3 Legal Warning banner 46
11.3.1 Background 46
11.3.2 Message of the Day 46
11.3.3 Telnet and SSH Banners 46
11.4 Basic pf Firewall Rules 47
11.4.1 Restricting Administrative Access 47
11.4.2 Limiting Outbound Traffic 47
11.5 Removing TCP Listener from X 47

12 Systems Administration and OpenBSD **49**
12.1 Patching OpenBSD 49
12.1.1 Patching Systems - Release, Stable or Current? 49
12.1.2 Binary Patches 50
12.2 Making Releases 50
12.2.1 Overview 50
12.2.2 Single Code Base Security? 50
12.3 Change Control 50

13 Criticism of OpenBSD **52**
13.1 IPv6 mbuf Patch Issue 52
13.2 Securelevels 52

14 Infrastructure Diversity and Security **54**

III Compliance, Financial Regulation and Control **55**

15 Overview **56**

16 Aligning Governance, Oversight and Regulation **57**

17 Mapping Requirements to OpenBSD **58**

18 Compliance Challenges **60**
18.1 Sarbanes-Oxley Act: 'Adequate Control'? 60
18.2 Approaching Risk 61

19 Open Source Software in Business **62**
19.1 Acceptance and Adoption 62
19.2 The Costs of Implementing Open Source Software 62
19.3 Continuity and Source Escrow 63

IV Case Study: Securing Treasury **64**

20 Background **65**
20.1 Confidentiality and Context 65
20.2 Overview 65
20.3 Security Assessment Drivers 66
20.4 Outsourcing, Offshoring and Risk 66
20.5 Treasury within the Wider Infrastructure 66
20.5.1 Segregation 66
20.5.2 High Security? 66
20.6 Conducting the Security Assessment 67

21 Findings **68**
21.1 Treasury Local Area Network 68
21.2 Server Configuration 69
21.3 User Authentication 70
21.4 Workstation Configuration 70
21.5 Time Synchronisation 71
21.6 Outdated Operating System Software 71
21.7 Firewall 71
21.8 Internet Downloads 71
21.9 Modems 71
21.10 Network Access Control 72
21.11 DNS Configuration 72
21.11.1 Overview 72
21.11.2 DNS Server 1 (Primary) 72
21.11.3 DNS Server 2 (Secondary) 73
21.11.4 Issues 73

22 Recommendations **75**
22.1 Network and Firewall 75
22.2 System Configuration and User Authentication 76
22.3 DNS 76

23 Designing a Secure Future Treasury **77**
23.1 The Challenge 77
23.2 Treasury Security Strategy 77
23.3 Technical Design 78
23.3.1 Network Diagram 78
23.3.2 Specification 78

V Conclusions and Looking to the Future **82**

24 Future Challenges in Compliance **83**

25 OpenBSD's Roadmap **84**

26 Conclusion **85**

Bibliography **86**

Appendices **93**

A Abbreviations **94**

Chapter 1

Executive Summary

This report aims to explore some of the security features of the OpenBSD computer operating system and how those features can be utilised to secure a Financially Sensitive Environment, such as a Treasury department for a large international financials firm.

OpenBSD is a collection of software developed by computer security enthusiasts in their spare time and made freely available (with all program source code) to the worldwide community under a relaxed licence. OpenBSD, like other computer operating systems such as Solaris, Windows and HP-UX, is loaded onto computer hardware and acts as the base on which business applications (for example, electronic funds transfer software) is run. OpenBSD contains a number of security features not found in other operating systems, and this report seeks to show how those features can provide a suitable level of security, whereby functionality can be tuned to an organisation's specific needs.

Financially Sensitive Environments (such as Treasuries and those dealing with funds transfer) are often subject to numerous statutory and regulatory compliance requirements (for example, PCI-DSS). Such requirements often involve the need to demonstrate that systems are secure against attacks on confidentiality, integrity and availability. OpenBSD's security features provide numerous benefits over other operating systems, by starting with a minimal configuration which is added to as necessary, rather than defaulting to activating all functionality which then requires systems administrators to remove the parts which are not needed. Additionally, OpenBSD's built-in network and application controls allow the operating system to define fine-grained controls over what business applications and users may do, increasing confidence that systems are only performing actions which are explicitly allowed by policy.

OpenBSD is not without its own issues, and these are discussed in each specific technology area. These issues raise the question about the overall management of security in the enterprise, and how systems and technologies are only one part of the security puzzle. The creation and maintenance of suitable, cost-effective policies, procedures, processes, guidelines and standards should be undertaken in concert with proper risk management across the enterprise, aligned with external obligations, business objectives and the capabilities of the corporate assets - infrastructure, personnel and business processes. OpenBSD can be used to increase the level of assurance in certain environments, provided that the application of the technology is done in cooperation with a security strategy which is driven by management support and effectively communicated to staff.

Part I

Introduction

Chapter 2

Introduction

2.1 Project Overview

This project is intended to show how the security features present within the OpenBSD operating system can be utilised to enhance the security of electronic data processing systems within a "Financially Sensitive Environment" (FSE) such as a corporate Treasury department. The project will first of all explore some of the major security features of OpenBSD, how they work, how they could apply to an FSE and any problems or challenges associated with those controls.

The project will then explore the wider compliance issues surrounding the application of security controls in such environments, including a discussion of some of the major regulatory and statutory areas, and how OpenBSD may help organisations deliver on their compliance responsibilities.

Following these background sections on technologies, issues and compliance, the project will then utilise a real-world case study showing some of the problems that an FSE may face, and how OpenBSD's tools can be applied to enforce the security requirements. This case study is based on a security assessment of a Treasury department within a large international financials firm.

The project concludes with a summary of the analysed features and how they can be aligned with the necessary control areas based on the investigations and recommendations which came out of the treasury security assessment case study. It also looks at the future for OpenBSD.

2.2 What is OpenBSD?

OpenBSD is a free, open-source, UNIX operating system which is driven by code correctness and security [43]. Unlike commercial operating systems such as Microsoft Windows [37], OpenBSD is free of any per-system or per-user licensing fees [39], and the source code to the operating system is provided so that anyone may make changes to it as they see fit.

Cryptography forms a major part of the OpenBSD platform [41], and the development team have included a number of tools to secure communications, including a full-featured IPSec implementation allowing OpenBSD to act as an IPSec gateway to provide secure communications between sites (tunnel), amongst other uses.

OpenBSD's audited code base has a reputation amongst security circles for having fixed many issues long before they become problematic in other operating systems. An example of this is where the OpenBSD developers had changed the algorithm in their fork of ISC BIND code, avoiding a recently published DNS cache poisoning attack [69].

2.3 What is a Financially Sensitive Environment?

A "Financially Sensitive Environment" (FSE) could be defined as one in which the financial operations of a company are undertaken. This could be accounting, treasury or an equivalent business function. It is not intended to mean an environment where there are severe budget constraints (i.e. price) however, as OpenBSD is free of licensing costs, this could well be an additional benefit. For the purpose of this project, a financially sensitive environment will be based on the concept of a Treasury department for a large multinational corporation. Each organisation may organise their accounting and financial control teams differently, but within the context of this paper an FSE is the department (the people, processes and systems) within the company that manages, reports upon and is responsible for cash control.

A Financially Sensitive Environment could be argued to be more than the direct controlling team over areas such as Accounts Receivable, Accounts Payable, Travel and Entertainment Expenses and so on. In fact, the reports generated by these teams are generally fed by data from outside their environment (i.e. stock control, sales, subsidiary, and partner fiduciary systems) so the dependency threshold could yet move back further to encompass systems and personnel feeding that data in to the primary target department. However, this is issue of scope definition, and although there is some merit

to envisaging the whole corporate entity as feeding into the financial data being controlled by an FSE, this must be carefully considered. This is especially important when evaluating scope for certification against standards such as ISO 27001, as certification is commonly undertaken against a well defined area of the business [77], with clearly indentified inputs and outputs. Failure to evaluate such scope properly can lead to unfeasibly large compliance and certification projects, which may become prohibitively expensive or ultimately unachieveable.

2.4 Project Objectives, Structure and Methodology

One of the main objectives of this project is to demonstrate how the security controls and concepts present within OpenBSD can be applied to financially-sensitive environments to provide additional levels of assurance and extra controls to ensure the Confidentiality, Integrity and Availability (CIA) of systems within such an environment. Additionally, the project will seek to explore the CIA augmentations, including Accountability, Authentication and Non-Repudiation. Where necessary, additional third party open source software packages will be introduced to contrast OpenBSD's capabilities or demonstrate additional control features not yet present within the base of OpenBSD.

It is the intention that the demonstrated controls and techniques can be applied to a financially sensitive environment so as to make a case for the consideration of OpenBSD against other more commercially established, closed source operating system platforms such as Sun Solaris and Microsoft Windows.

To test the effectiveness of some of the security features present with OpenBSD, several scenarios will be enacted based on the security control in question to illicit the control response and evaluate if the control has performed as expected, and if that control is aligned with the needs of the system in that context. For example, execution enforcement controls such as systrace will be tested by inserting a program into the environment, a policy defined and the application tested to see what can be achieved outside of the expected (and policy enforced) functionality. For network elements (for example, the pf firewall), some open source network scanning tools such as nmap [19] and other selected penetration test utilities will be applied to demonstrate the level of robustness and provide evidence of the effectiveness of that specific control.

2.5 Document Production

This document was written in LaTeXusing vim on OpenBSD 4.1-STABLE. Diagrams were produced in Microsoft Visio 2007.

Part II

OpenBSD: Technologies and Features

Chapter 3

Overview

This part of the project provides an overview of OpenBSD, its history and what technologies and security features are present within the operating system. This project looks at some of the major security features and then explores how those features are applied in practice:

- What is the security control?
- How does the security control work?
- How can the security control be applied in the context of an FSE?
- Are there problems with the security control, and if so, how can they be mitigated/compensated?

For pf firewall and systrace, an evaluation of the control's effectiveness is provided within an FSE context. This part of the project will also look at some of the other features of OpenBSD which may be useful in the context of a secure environment, explore some of the criticism of OpenBSD and its developers, describe steps to harden the default installation of the operating system and cover some of the other related issues and challenges.

Chapter 4

OpenBSD History, Licensing and Source Access

4.1 History

OpenBSD is a UNIX-like open-source operating system which is developed by a team of volunteers and made available free of charge. "BSD" refers to the heritage of the Berkeley Software Distribution version of UNIX from which it is derived. BSD originates from the University of Berkeley, California, USA where it was first forked from an original AT&T UNIX tape distribution in the late 1970s by two Berkeley students, Bill Joy and Chuck Haley [20, page 14]. Over the next 35 years, BSD has been continually developed and split into a number of different flavours, each with their own focus and aims. Examples of these flavours (and key objectives are) [20, page 20]:

- FreeBSD - Goal is making BSD easy to use and support many ported applications.
- NetBSD - Aims to support as many hardware platforms as possible.
- OpenBSD - Focus on code quality, auditing and security.

The OpenBSD project was founded by Theo de Raadt in October 1995 when he forked the NetBSD code base [63, page 9] following several strong disagreements with the rest of the NetBSD development team [34, page 4]. The first public release of OpenBSD was version 2.0, which was made available in mid-1997.

Today, OpenBSD is still under active development. Currently at release 4.1, the developers release a new version approximately every six months [21]. OpenBSD supports a number of processor architectures, including Intel

x86, Sun SPARC, DEC Alpha and VAX [59]. OpenBSD's diverse platform support comes in part from the original NetBSD fork and Theo de Raadt's belief that identifying bugs in code is easier when that code is being run on a number of different platforms and fixes are applied upstream when problems are discovered in one area [73]. According to a survey conducted in 2005 by the BSD Certification Group, OpenBSD accounts for approximately one third of all *BSD installations [4].

4.2 Licensing

Whilst debating the merits of different licensing schemes is outside the scope of this project, it is worth mentioning that the licensing model utilised for OpenBSD is different to that of more widely recognised Free, Open Source Software (FOSS) products such as Linux. OpenBSD is released under the Berkeley Software Distribution (BSD) Licence, which originates from the University of Berkeley, California, USA. It is important to note that some non-BSD licensed code such as the gcc compiler (GNU GPL Licence) and Apache web server (Apache Licence) is included in the OpenBSD base. The BSD licence allows for commercial companies to take a copy of the code and extend (or modify) the product then sell the resulting binaries without having to provide the source code. This is in contrast to the GNU Public Licence (GPL) which covers software such as Linux which requires that any changes be made available in source form if changed binaries are released. Several commercial vendors [40] use OpenBSD as the base of their products.

Licensing is taken extremely seriously by the OpenBSD developers, even to the extent that a minor change to the licence in a ported application (or any ambiguity in such) can result in its complete removal from the OpenBSD ports and packages collection [11].

The fact that OpenBSD (like the other BSDs) is under the Berkeley licence means that the deployment of the software (whether personal, non-profit or commercial) incurs no licensing fees, regardless of the number of users on the system, processors or concurrent processes. This is a stark contrast to most closed-source commercial operating systems (e.g. Microsoft Windows), where individual licences may be required for each workstation, user and server [39].

4.3 Accessing the OpenBSD Source Code

In 1995, Chuck Cranor and Theo de Raadt deployed the very first "anonymous" open source CVS (Concurrent Versioning System) server at the Washington University in St. Louis, Missouri, USA. This system allowed anyone

with Internet access to browse the source code repository for the OpenBSD project, and be able to see changes between versions of code. With OpenBSD's security goals in-mind, the CVS repository itself was designed to be secure, with code updates being "pushed" or sent from the Master CVS server (under de Raadt's control) to the remote Anonymous CVS repositories, from where anonymous users could then "pull" or download the code to their own machines. This effectively enforces a one-way data flow, as anonymous users have no direct access to the Master CVS server and therefore cannot modify the code as no changes are pulled back to the Master from the satellite repository systems [12]. However, if an attacker were to be able to compromise one of the anonymous CVS repositories and inject malicious code, then all other users downloading code from that repository would collect the modified versions and could be susceptible to compiling code which would allow the attacker to access their systems or perform any other task of their choice, especially when attacking the source code to the gcc compiler. An excellent example of such an attack was explored in Ken Thompson's classic paper on inserting backdoors into compiler code [78].

It is now common practice for open source projects to feature an online, anonymously accessible read-only CVS server, and many multi-project open-source community sites (such as Sourceforge.net) have sprung up to deliver such functionality to small projects [35], many of which may not have the infrastructure or money to run their own CVS servers.

The OpenBSD development team has decided that the stale, unmaintained GNU CVS code is overly complex, and is being rewritten from scratch as part of the BSD-licensed OpenCVS project [33]. Over time the OpenBSD project's CVS repository will be migrated to the new versioning system, and all OpenBSD developers will work on the system source code via the OpenCVS repository systems. OpenCVS is still under development and no timescales have yet been set for the migration.

Chapter 5

The pf Firewall

5.1 What is pf?

The pf (packet filter) firewall is a BSD licensed piece of software developed by Daniel Hartmeier for the OpenBSD project in 2001. It replaced the previous firewall software "IPFilter" owing to licence concerns about redistribution. The creator of IPFilter, Darren Reed, had included several licensing stipulations in certain pieces of his software; the OpenBSD team thought it best to remove his software from OpenBSD and develop their own solution [10]. The pf firewall is a stateful packet filter which provides access control checking to network packets arriving on or leaving from the system running pf. A firewall such as pf is configured through the use of a policy which specifies how Internet Protocol services are to be handled. Actions which the system can take include dropping the packet (denying), forwarding the packet (allowing) or proxying the packet through a intermediary control function to validate packet contents. Not only can pf handle packets to and from itself (i.e. a host-based firewall [65, page 296]), but also can act as a filter between machines on either side of two network interfaces, mediating access between the two and enforcing the specified policy. This can happen either as a routed interception (whereby the pf firewall is a specified hop in the route) or can be a transparent bridge whereby packets are silently intercepted through the inbound network interface and inspected before being passed out through the outbound network interface [79].

Support for various protocols makes pf extremely flexible. At the time of writing, pf supports both IPv4 and IPv6, which allows system administrators to define very explicit rule sets, with one system able to handle and inspect traffic to/from mixed IPv4 and IPv6 networks. Additionally, OpenBSD can also act as an IPv4-IPv6 gateway, through use of the gif(4) interface [44]. Network Address Translation (NAT) is supported natively in pf, as is IP Quality of Service (QoS) via the ALTQ priority mapping.

The pf firewall is one of many different firewall products available on the market [16]. IPFilter continues to be developed by its author, and in addition there are many commercial closed-source firewalls including Checkpoint Firewall-1 (running on Microsoft Windows, Sun Solaris and the Nokia IPSO platforms) and Cisco Pix.

5.2 Using pf in an FSE security context

The security properties of pf firewall are key elements to restricting network access to systems to only those permitted by policy. As will be seen in "Part III - Compliance, Financial Regulation and Control", pf is key to delivering upon a number of compliance requirements, especially for those laid down in the protection of financially sensitive environments - an example of this is "Requirement 1: Install and maintain a firewall configuration to protect cardholder data" of the PCI-DSS [64]. In "Part IV - Case Study: Securing Treasury", we will look at how a real-world FSE can utilise the pf firewall to partition network zones into different security segments and define policies to control communication between them.

5.2.1 Ubiquitous Technology?

According to [15], the use of firewalls is now almost universal (90-97%) throughout most large businesses. FSEs especially would benefit from the use of such technology, as setting appropriate firewall policies enables enforcement of network level controls over which machines may communicate, in which directions and with what services.

5.2.2 Auditing Connections

In addition, pf supports logging to a variety of interfaces, including syslog [57]. This can provide an audit trail of communications depending on the type of detail logged, and to which packets logging is applied. If every connection was to be audited, each would generate a log record complete with timestamp and other pertinent information - this could generate extremely high volumes of records and therefore require much disk space. This would have additional lead-on impact, such as performance of search queries and ability to securely backup the data. Organisations may wish to apply restricted logging to only certain types of traffic depending on their risk profile and business requirements, for example only logging traffic which is prohibited by policy (denied packets).

5.2.3 Data Flow Enforcement

As mentioned previously, the use of a firewall technology allows organisations to stipulate inbound and outbound policies for network traffic, including which services. An FSE may make use of such technology in order to apply a layered security model to its network structure, constraining certain applications within a semi-public zone to prevent direct access to sensitive core systems. OpenBSD's pf allows for configuration of many aliases, tables and groupings for systems, network segments and service types, for ease of administration and configuration [56]. An FSE could use a number of physical interfaces on the firewall system to configure Demilitarised Zones (DMZs) to setup semi-public network segments in which to host accessible services such as web servers and mail servers. These will then be constrained outside of sensitive network areas or core network to limit an attacker's access if one of these services were to be compromised. By setting a restrictive traffic policy between network zones, it would then require an attacker to find a path between zones in order to compromise additional systems outside of the DMZ.

5.2.4 Firewall Administration

Configuration files for pf are plain-text files [47]. Central administration could be securely and cheaply achieved by using other built-in tools within OpenBSD. A central firewall administration workstation could be used to create the policy files, and this system could then copy the policy files to the target firewall systems using a secure method such as scp or sftp (Secure Shell (SSH) based secure copy/file transfer mechanisms). Access to the SSH port on the target firewalls could be restricted to the admin workstation in the configuration files of pf, sshd or both.

5.2.5 High Availability

OpenBSD's pf firewall also features Common Address Redundancy Protocol (CARP) [45] and pfsync [48] which allows for failover of firewall systems to enable high availability. Should the primary firewall fail, the backup firewall will detect this and transfer all states and connections to itself using the heartbeat interface which connects the two firewalls together directly, usually by a cross-over cable between them. Such functionality is also available on commercial firewalls, but commonly is only available as an extra module at significant extra licensing cost [38]. As with the rest of OpenBSD, both pf (and by extension, CARP, pfsync and all other tools) are free. These features could also possibly be used to provide high availability functionality to other applications (for example, a financial application server). The hoststated(8) daemon has been developed to link in with pf to provide load balancing functionality, to split traffic loads between multiple hosts [46].

5.3 Problems with pf

At the time of writing there are no published bugs or acknowledged security issues with pf directly. It is worth noting that the pf filter and NAT rule files can be set to immutable by raising the kernel's securelevel to 2 [50]. However, as discussed in the section "Criticism of OpenBSD", it has recently been discovered that there are security problems with securelevel(7), and continued research into the software may find additional issues which could affect pf's security when securelevels are used. The use of securelevel(7) could lead to a false sense of security in this sense, so proper risk management is required.

5.4 pf - an example in practice

In order to demonstrate the effectiveness of the pf firewall, the following example has been constructed which shows a real-world scenario where the control may be applied in an FSE. In this instance, an OpenBSD system running the build-in ISC BIND server is used and proves that the filtering capability effectively stops the network access restricted by the pf rule set. Domain Name Service (DNS, resolving system host names to IP addresses) is especially important for an FSE, as without proper security controls surrounding it, legitimate traffic may be redirected (DNS Poisoning) to malicious "phishing" sites where sensitive information may be captured. A Sun Netra X1 rackmount server running OpenBSD 4.1/sparc64 is the target machine - the two services accessible are the SSH daemon (sshd, running on 22/tcp) and BIND (named, running on 53/tcp+udp). A simple pf rule set has been defined, which permits DNS queries from any system on the local network, and restricts SSH access to one approved administrative workstation (10.0.0.1).

5.4.1 System Services

To show the services are running on the target machine (10.0.0.252), the following commands were executed as root:

TCP services listening: `# netstat -an | grep LISTEN`

```
tcp        0      0  *.22                   *.*        LISTEN
tcp        0      0  10.0.0.252.53          *.*        LISTEN
```

UDP services listening: `# netstat -an | grep 53 | grep udp`

```
udp        0      0  10.0.0.252.53          *.*
```

5.4.2 Target machine pf rule set

The following rulset is configured on the target machine (10.0.0.252) within `/etc/pf.conf`.

```
# Macros
int_if  = "dc0"
lan_net = "10.0.0.0/24"

# Options
set block-policy return
set loginterface $int_if
set skip on lo

# Scrub
scrub in all

# Default Deny Policy
block inet6
block in all
pass out

# activate spoofing protection for all interfaces
block in quick from urpf-failed

# Allow SSH from Admin station only
pass in quick on $int_if proto tcp from 10.0.0.1 to $int_if port
     ssh

# Allow NTP Requests to this server
pass in on $int_if proto udp from $lan_net to any port 123

# Allow DNS Requests to this server
pass in on $int_if proto { tcp, udp } from $lan_net port
    1024:65535 to any port 53

# Allow DHCP Requests to this server
pass in on $int_if proto udp from any port 67 to any port 68
```

5.4.3 Results with pf enabled

This section shows the results of various tests with pf enabled and it filtering packets. In this first scenario, PF is enabled and utilising the rules defined in the rules shown above.

nmap scan The results from the nmap scan from system 10.0.0.24 show that only port 53 is accessible as it is not the approved administrative workstation (10.0.0.1) the SSH port is not visible.

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at
    2007-07-27 13:34 BST
```

```
Interesting ports on hades.willowcottage.net (10.0.0.252):
Not shown: 1679 closed ports
PORT    STATE SERVICE VERSION
53/tcp open  domain  ISC Bind 9.X
MAC Address: 00:03:BA:10:6B:09 (Sun Microsystems)
Device type: general purpose
Running: OpenBSD 3.X
OS details: OpenBSD 3.5 or 3.6

Nmap finished: 1 IP address (1 host up) scanned in 9.576 seconds
```

DNS Resolution The following test shows the DNS resolution attempt from 10.0.0.24 resolving OK against the target system using `# dig www.google.com @10.0.0.252`

```
; <<>> DiG 9.3.4 <<>> www.google.com @10.0.0.252
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16300
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 6,
    ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         70993   IN      CNAME   www.l.google.com
    .
www.l.google.com.       272     IN      A       216.239.59.147
www.l.google.com.       272     IN      A       216.239.59.99
www.l.google.com.       272     IN      A       216.239.59.103
www.l.google.com.       272     IN      A       216.239.59.104

;; AUTHORITY SECTION:
l.google.com.           76931   IN      NS      g.l.google.com.
l.google.com.           76931   IN      NS      a.l.google.com.
l.google.com.           76931   IN      NS      b.l.google.com.
l.google.com.           76931   IN      NS      c.l.google.com.
l.google.com.           76931   IN      NS      e.l.google.com.
l.google.com.           76931   IN      NS      f.l.google.com.

;; Query time: 6 msec
;; SERVER: 10.0.0.252#53(10.0.0.252)
;; WHEN: Mon Aug 27 14:33:47 2007
;; MSG SIZE  rcvd: 212
```

SSH Connection This test shows the failed attempt to connect from 10.0.0.24 to the DNS server's SSH port using `# ssh 10.0.0.252`. This is refused as the originator (10.0.0.24) is denied from connecting to 22/tcp by

the target's pf configuration.

```
ssh: connect to host 10.0.0.252 port 22: Connection refused
```

5.4.4 Results with pf disabled

This section shows the results of various tests with pf disabled and no packet filtering.

nmap Scan The results from the nmap scan from system 10.0.0.24 show that both ports 53 and 22 are accessible.

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at
    2007-07-27 13:39 BST
Interesting ports on hades.willowcottage.net (10.0.0.252):
Not shown: 1678 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 4.6 (protocol 2.0)
53/tcp open  domain  ISC Bind 9.X
MAC Address: 00:03:BA:10:6B:09 (Sun Microsystems)
Device type: general purpose
Running: OpenBSD 3.X
OS details: OpenBSD 3.5 - 3.9

Nmap finished: 1 IP address (1 host up) scanned in 26.321
    seconds
```

DNS Resolution As with the previous test, the following shows the DNS resolution attempt resolving OK against the target system from 10.0.0.24 using `# dig www.google.com @10.0.0.252` - this is owing to the fact that pf has been disabled and no packet filtering is happening.

```
; <<>> DiG 9.3.4 <<>> www.google.com @10.0.0.252
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16300
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 6,
    ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         70993   IN      CNAME   www.l.google.com
    .
www.l.google.com.       272     IN      A       216.239.59.147
www.l.google.com.       272     IN      A       216.239.59.99
```

```
www.l.google.com.       272     IN      A       216.239.59.103
www.l.google.com.       272     IN      A       216.239.59.104

;; AUTHORITY SECTION:
l.google.com.           76931   IN      NS      g.l.google.com.
l.google.com.           76931   IN      NS      a.l.google.com.
l.google.com.           76931   IN      NS      b.l.google.com.
l.google.com.           76931   IN      NS      c.l.google.com.
l.google.com.           76931   IN      NS      e.l.google.com.
l.google.com.           76931   IN      NS      f.l.google.com.

;; Query time: 6 msec
;; SERVER: 10.0.0.252#53(10.0.0.252)
;; WHEN: Mon Aug 27 14:33:47 2007
;; MSG SIZE  rcvd: 212
```

SSH Connection This test shows the successful attempt to connect from 10.0.0.24 to the DNS server's SSH port using `# ssh 10.0.0.252`. This is accepted as the SSH port is no longer restricted by pf.

```
**************************************************************
hades.willowcottage.net DNS Server
Authorised users only
**************************************************************
root@10.0.0.252's␣␣password:
```

Chapter 6

OpenSSH

6.1 What is OpenSSH?

OpenSSH is a sub-project of OpenBSD. It is a free, open-source collection of Secure SHell (SSH) tools providing encrypted replacements for traditional UNIX utilities such as telnet and ftp [60]. OpenSSH supports both versions 1 and 2 of the SSH protocol standards as defined within [25] and [24]. The problem with traditional utilities such as Telnet is that the login details and subsequent command and reply traffic traverses the network in the clear (plain-text), rendering the stream susceptible to unauthorised reading and/or modification. The OpenSSH tools mitigate these issues by utilising secure protocols based on cryptography, with SSH supporting mechanisms such as public-key based authentication. OpenSSH has become extremely popular, and has been integrated into many other open-source platforms (including Linux) and even proprietary systems and devices such as those manufactured by Nokia, Cisco and HP [61].

6.2 Using OpenSSH in an FSE security context

The use of SSH tools instead of traditional insecure protocols such as Telnet brings benefits to Financially Sensitive Environments in terms of providing a degree of assurance over the security of network traffic between systems. Through encryption of data, a malicious entity could no longer view plain text intercepted data streams, and through proper validation of host keys, systems can confirm that the system they are talking to is authentic. The use of SFTP (Secure Shell FTP [26]) instead of plain text FTP also brings significant security benefits, as the login exchange and subsequent data transfer would be secured by encryption, providing control over both confidentiality and integrity [2]. An auditor assessing communications between systems could then be confident that host-host traffic was secured and no longer liable to potential unauthorised viewing or alteration. By using secure pro-

tocols instead of insecure ones [65, page 292], problems such as TCP/IP session hijacking can also be mitigated [17, page 156].

OpenSSH provides a useful set of utilities for administration, file transfer and tunneling other insecure protocols, building confidentiality and integrity into communications. In "Part III - Compliance, Financial Regulation and Control", we shall see that the need to have assurances over the security of data in transit can be mapped to OpenSSH's abilities. The future secure Treasury design described in "Part IV - Case Study: Securing Treasury" will show how OpenSSH is a key element in enabling secured administration of servers, and providing a secure replacement for the Treasury's previously insecure file transfers using FTP.

6.3 Problems with OpenSSH

SSH, like other protocols, is susceptible to a "man in the middle" attack, and if an attacker were able to fool administrators that their SSH host key was that of the intended target system, then data could be passed to a malicious third party [2, page 79]. Such an attack can be mitigated by putting in place proper procedures to validate key fingerprints. A way to approach this for each system administrator to make note of their valid key fingerprint, and then confirm this with one another over an out-of-band channel such as by voice telephone conversation. This in itself requires that the administrators know and trust each other, and there should be organisational policies in place with a management approved method for such validation. SSH will flag an alert on connection if the host key is either unknown or has changed since the host keys were input, which would alert an administrator that something had gone awry. Such an incident could indeed be benign, as a previously valid system which had been upgraded or moved to a new physical server would generate a different host key the first time the SSH daemon was invoked [2, page 17]. This is an excellent example of 'Security as a Process' and it demonstrates the importance of overall security strategy and wrap-around policies/procedures to cover management oversight of security across the enterprise.

Chapter 7

Systrace System Policies

7.1 What is systrace?

One of the more flexible tools in the OpenBSD system is Systrace. Systrace was written by Niels Provos and has been part of the base system since OpenBSD version 3.2 [54]. Systrace "monitors and controls an application's access to the system by enforcing access policies for system calls" [53].

Systrace allows system administrators to define a security policy for each individual application executed on the system, from reading and writing files to controlling even if the code may make a network connection and then to what specific host(s). These policy elements can include fine grained control over execution context of the code, for example only allowing applications to perform sensitive functions if executed by a user with the appropriate group membership. Such control even supersedes the all powerful 'root' (UID 0) user, which traditionally has complete control within a UNIX environment.

By generating, fine tuning and enforcing Systrace policies, additional constraints can be applied to applications above and beyond the level of control afforded by traditional UNIX control mechanisms such as permission masks, group membership and firewall configuration.

A good example of Systrace's flexibility can be demonstrated in its ability to constrain an application to only be able to read certain files and write to certain files, all whilst preventing that application from sending or receiving any network traffic whatsoever. Such a set of controls may be extremely useful in providing additional security barriers for third party software, some of which may be emulated, closed-source binaries which cannot be directly audited as their code is not publicly available for review.

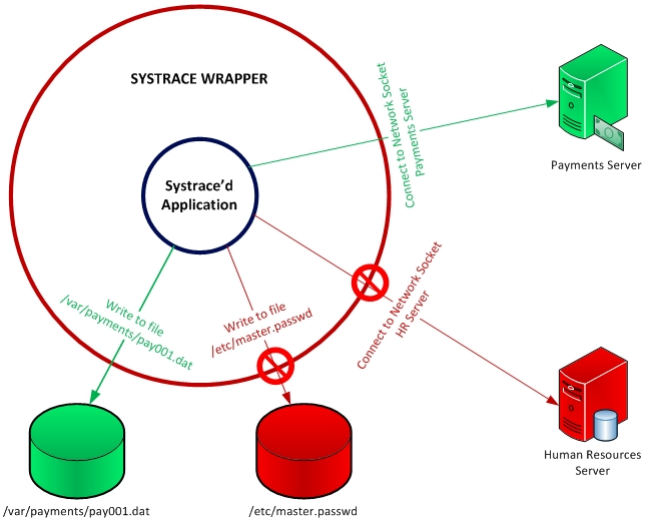
One of the elements of systrace is the writing of policies for use by the

systrace binary in enforcing controls over the system calls. Systrace has an automatic generation function (systrace switch -A) whereby an application can be invoked inside the Systrace wrapper and a policy file generated automatically as the application is used. This essentially records the system calls being used, and no controls are applied during the policy generation process. The output of this automated 'training mode' results in a policy baseline from which duplicate calls can be removed and then the security officer (or system administrator) can then verify each action as either being acceptable or not. This then allows the tuning of the policy file, whereby specific calls can be approved or rejected by use of a fixed policy syntax, as defined within the systrace manual pages [54].

7.2 Using systrace in an FSE security context

Systrace's power derives from its ability to define extremely fine grained permissions over what executing code may do in the form of system calls. This means that an application can be given only the privileges it needs to get the job done (Least Privilege). The reality of this ability in an FSE context is that applications can be constrained to do only their approved functions, and these functions could then be split out into several roles. As described in the Treasury Security Assessment (Chapter 21), one of the major issues was the world-writable access to plain-text payments files on the Electronic Funds Transfer (EFT) server. Although proper UNIX permission masks, firewalls and cryptography are part of a solution to adequately addressing such an issue, the addition of tools like Systrace could complete the end-to-end security picture by restraining the payments application - for example, preventing the writing of any sensitive data outside of pre-approved directories and only allow the application to transmit payment data to the approved recipient systems as defined within the systrace rules for the payments application.

In "Part III - Compliance, Financial Regulation and Control", we will explore how the application contraints which can be imposed by Systrace map to compliance and best practice controls, for example "Control 11.2.2 - Privilege Management" in ISO 17799:2005 [28]. In "Part IV - Case Study: Securing Treasury", we will see how the use of Systrace in Treasury could enhance system integrity by enforcing least privilege for third-party financial applications which could be compromised.



7.3 Problems with systrace

Systrace should not be considered a magical solution to application execution security, as it is not without its own security faults. Whilst writing this project, a major new flaw has been discovered in systrace, whereby processing concurrency can be exploited to allow an attacker to escalate his privileges and bypass auditing [81]. This flaw is so severe that the authors of the popular SysJail virtualisation environment have placed a notice on the front page of their website urging all users of SysJail and Systrace to discontinue using the tools until further notice [76].

7.4 Systrace - an example in practice

7.4.1 A Simple Payments Processing Program

To demonstrate the effectiveness of systrace, an example has been created to explore policy file creation, modification and subsequent invocation. To start with, a directory was created for testing a simple shell script and the script generated within it, plus some directories against which the policy could be tuned. The `/var/test` directory contains two sub-directories:

/var/test/hr	Human Resources Records
/var/test/payments	Payment files

In this example, the shell script will act in place of a payment server application that should only be able to read files from the payments directory. It should not be able to read anything from the human resources directory. In order to show how userid is irrelevant for systrace, the program will be

invoked as the root user. The shell script (`/var/test/myprog.sh`) is as follows:

```
#!/bin/sh
cat payments/pay001.dat
cat hr/employees.txt
```

7.4.2 Generating a Systrace Policy

To generate the initial systrace policy file, the program was invoked as `# systrace -d /var/test -A ./myprog.sh` - the output produced from this command is:

```
BIC0001 DGHN56  GBP      130098729.11     2016778 18-02-90

BLOGGS, J.      18-09-76        372728  London
FOX, M.         16-02-83        637281  Maidstone
```

The output above shows the test script echoing the payment file and then the Human Resources file. This is owing to the fact that Systrace is in learning mode, and the whole program runs as normal without any contraints. The resulting policy file was placed into the `/var/test directory` as `var_test_myprog_sh`. The policy file produced is as follows:

```
Policy: /var/test/myprog.sh, Emulation: native
        native-__sysctl: permit
        native-mmap: prot eq "PROT_READ|PROT_WRITE" then permit
        native-mprotect: prot eq "PROT_READ" then permit
        native-fsread: filename eq "/etc/malloc.conf" then
            permit
        native-issetugid: permit
        native-sigaction: permit
        native-getpid: permit
        native-__getcwd: permit
        native-getppid: permit
        native-gettimeofday: permit
        native-geteuid: permit
        native-getuid: permit
        native-getgid: permit
        native-getegid: permit
        native-fsread: filename eq "/var/test/myprog.sh" then
            permit
        native-fcntl: cmd eq "F_DUPFD" then permit
        native-close: permit
        native-fcntl: cmd eq "F_SETFD" then permit
        native-fstat: permit
        native-sigprocmask: permit
        native-read: permit
        native-fsread: filename eq "/sbin/cat" then permit
        native-fsread: filename eq "/usr/sbin/cat" then permit
        native-fsread: filename eq "/bin/cat" then permit
        native-fork: permit
```

```
native-sigsuspend: permit
native-execve: filename eq "/bin/cat" and argv eq "cat␣
    payments/pay001.dat" then permit
native-getrusage: permit
native-wait4: permit
native-sigreturn: permit
native-execve: filename eq "/bin/cat" and argv eq "cat␣
    hr/employees.txt" then permit
native-munmap: permit
native-exit: permit
```

7.4.3 Modifying the Systrace Policy

This automatically created policy (as it stands) allows the program to "cat" the contents of the data files to the terminal - we now need to restrict this so that only the payments file can be read by the payments application test script. By removing the native-execve entry for the `hr/employees.txt` file and saving the modified policy, we can now re-run the systraced program using the modified systrace policy `# systrace -a -d /var/test -E /var/test/systrace.log ./myprog.sh`. The resulting output from the program running wrapped in the modified policy is:

```
BIC0001 DGHN56  GBP      130098729.11     2016778 18-02-90
```

The script has now been unable to access the HR file. To verify that access to the file has been denied by systrace policy, we can review the log file (generated with the systrace switch -E) at `/var/test/systrace.log`:

```
systrace: deny user: root, prog: /var/test/myprog.sh, pid:
    22302(0)[9042], policy: /var/test/myprog.sh, filters: 33,
    syscall: native-execve(59), filename: /bin/cat, argv: cat hr/
    employees.txt
systrace: deny user: root, prog: /var/test/myprog.sh, pid:
    22302(0)[9042], policy: /var/test/myprog.sh, filters: 33,
    syscall: native-fsread(5), filename: /usr/share/nls/C/libc.
    cat
systrace: deny user: root, prog: /var/test/myprog.sh, pid:
    22302(0)[9042], policy: /var/test/myprog.sh, filters: 33,
    syscall: native-fsread(5), filename: /<non-existent filename
    >: /usr/share/nls/libc/C
```

This test shows that the reading of files has been restricted to only those permitted within the systrace policy. More complex policy files can restrict access to network resources, system memory, file access modes (read, write, execute) or any other operation performed via a system call.

The next test is to simulate an attacker having been successful in inserting malicious code into the payments application. In this case, we shall

insert the command `cat /etc/master.passwd` into the test script, which would display the contents of the shadow password file containing the encrypted password hashes. As the application script is running as root, it would by default have access to this file. If we re-run the test script again, the output remains the same:

```
BIC0001 DGHN56  GBP      130098729.11     2016778 18-02-90
```

To prove that the access was intercepted and denied by policy, we can view the resulting `systrace.log` file entry:

```
systrace: deny user: root, prog: /var/test/myprog.sh, pid:
    15791(0)[27203], policy: /var/test/myprog.sh, filters: 33,
    syscall: native-execve(59), filename: /bin/cat, argv: cat /
    etc/master.passwd
```

Such restrictions prove that systrace has the capability to constrain the actions of modified programs, even when run in the security context of the most powerful user (root) on the system.

Chapter 8

Memory Protection Features

8.1 Features within OpenBSD

8.1.1 W^X

The W^X feature helps make exploitation of certain software bugs more difficult by setting a portion of memory either Writable OR eXecutable. In this way, a portion of memory to which an application may write can then not also be executable, which means if unauthorised instructions are written into that portion of memory, that the particular area of memory cannot then be subsequently executed. On some hardware architectures, this feature is supported in hardware (such as sparc, sparc64 and alpha), whereas on other architectures such as i386 the feature requires binary changes [8, page 5]. An example of this is if a stack overflow [36, page 290] is found in an application and an attacker can insert a privilege command [71, page 565], but with W^X this code stack may have been modified but the contents would not be executable.

8.1.2 .rodata Segment

This is where program constants, strings and pointers, are stored in a read-only segment to prevent any run-time modification of constants being executed [8, page 13]. The developers intended that the address spaces have a minimal set of permissions (Least Privilege).

8.1.3 Randomized malloc() and mmap()

This feature randomises the address spaces used by an application each time it is run, mitigating exploits against fixed memory mapped spaces [7, page 22]. As the application would use a different memory allocation each time, a fixed address exploit would be unlikely to set to the right address location, making that exploit unworkable.

8.1.4 strlcpy() and strlcat()

Ubiquitous C string functions such as strcpy() and strcat() (copy and concatenate) have been replaced in recent years with strncpy() and strncat(), which now require that the length of the string be defined. However, strncat() and strncpy() require that the programmer understand some detailed nuances, and also the two functions are unable to detect truncation. In response to this problem, Theo de Raadt and Todd C. Miller created the strlcpy() and strlcat() functions which *"guarantee to NUL-terminate the destination string for all strings where the given size is non-zero"* [13]. These new functions give programmers an easier way to handle fixed length buffers and reduce the potential for overflows which can then be exploited to gain access or raise privilege with injected code.

8.2 Using memory protection in an FSE security context

Most of the memory protection features within OpenBSD could be viewed as an indirect (and in some cases, transparent) benefit to using the operating system. With some of the key controls based within the kernel, the benefits are derived simply from using applications on the operating system and without any user or administrator involvement. With regards to security within an FSE, these additional controls could be regarded as compensating or mitigating controls for potentially problematic applications. OpenBSD's ability to emulate other UNIX systems (such as HP-UX) means that an application could be hosted on OpenBSD with binary emulation, and with security features such as memory protection, the pf firewall and systrace, allow for a much tighter control over the execution of that business application. In using OpenBSD to host financially sensitive applications, auditors could be given a higher degree of confidence over the execution security of those applications, as they would be subject to constraints which make exploitation by attackers that much more difficult, such as the injection of shellcode [17, page 84], controls which may not be available on the application's native Operating System.

8.3 Problems with memory protection features

One of the key challenges with memory protection is that for some of the controls they must be actively utilised by programmers in the creation of software. This is certainly evident in third-party software which is present in the OpenBSD ports tree, as the ports maintainers often have to create patch files in order to modify the base code of a ported application to use safer functions like strlcpy() instead of strncpy() [13]. Such activities require

careful code auditing to make sure that the unsafe functions are captured and corrected, either by manual code review, the use of file scanning tools or a combination of both. For larger software projects in the OpenBSD ports/packages collection (such as OpenOffice or Firefox) which run into tens or hundreds of thousands of lines of code, such a task can be a significant investment of time and resources.

Chapter 9

Auditing

9.1 Auditing within OpenBSD

9.1.1 syslog

As with most implementations of UNIX, OpenBSD includes a syslog daemon which captures and records messages generated by the system and applications [52]. These records can retained in a number of ways, from writing to disk-based files, sending the messages to a remote syslog server, outputting to physical printer or a combination of those methods. These can be tuned depending on the security requirements of the environment, and syslog levels can be redefined to allow efficient capture of log messages to support wider infrastructure monitoring solutions.

In "Part III - Compliance, Financial Regulation and Control", we will see how auditing and the proper management of logs maps to compliance and best practice controls, including Section "10.10 Monitoring" in ISO 17799:2005 [28] and "Requirement 10: Track and monitor all access to network resources and cardholder data" in PCI-DSS [64]. In "Part IV - Case Study: Securing Treasury", we shall see how OpenBSD's logging features can be applied to an FSE in order to meet requirements and provide a trail of evidence for financial processing.

9.2 What additional auditing features are available?

9.2.1 Output to Physical Log Printer

OpenBSD's syslog implementation allows for the printing of syslog messages as they arrive. This may be an especially useful control for jurisdictions where hard copy evidence is required for prosecution of system intruders.

An example of UNIX logs being output to hard-copy paper trail was demonstrated in [74], where the terminal traffic was copied to physical printers for use as evidence in the case of KGB hacker Markus Hess, as he infiltrated US universities and military computer networks. OpenBSD can be configured to log to the physical printer by setting `/etc/syslog.conf` to redirect messages to the physical printer device (such as `/dev/lpt0`).

9.2.2 Logging to Write-Once, Read-Many Media

If logs are sitting on a server in an electronic format then they could be susceptible to modification. By placing logs onto CD-Recordable or DVD-Recordable media, then a write-once copy of the log files would then be placed onto optical media which cannot be modified. Owing to the nature of technology, it is not possible to write message-by-message records to the disc, which means that a buffer of messages would have to be stored on the logging system and could then be susceptible to in-buffer modification before the logs were written onto the disc.

9.3 Using auditing in an FSE security context

Proper audit trails are a core requirement of most financial environments, and many of the standards and regulations for these industries mandate proper log controls, for example PCI-DSS [64, Requirement 10] and the UK Financial Services Authority controls [18, SYSC]. The generation of audit trails enables financially sensitive environments to be able to trace transactions and activities, which is crucial in financial transactions not only to support correctness and aid in finding transaction problems, but also to discourage (and discover where necessary) fraud by employees, partners and suppliers who may have access to payment or accounting systems.

9.4 Problems with auditing

9.4.1 Physical Logs Under Attack

In [20, page 670] it is recommended that such physical output be sent to a dedicated progressive printer (such as a dot matrix) as output to a page printer would be cached until a whole pages worth of data was ready to be printed, and as such the stored print buffer could be tampered with by the attacker to alter the output or even to prevent the physical output altogether. In addition, once an intruder has gained a privileged level of access to the system, they could disable the logging or permanently modify logging daemons to mask their activity. However, even a progressive printer could be attacked directly to damage the physical output the attacker could reverse feed the print paper to overwrite or modify the printed log data,

rendering the physical copies potentially inadmissible in court by introducing doubt as to their integrity. In a single page output buffer print scenario, such an attack would not be possible as the physical page would be detached and dropped into the output bin, requiring physical presence to feed the sheet back into the printer (or steal it!), however, any buffered page output could be attacked and subject to modification or erasure. By dedicating a printer to physical log output and forbidding its use for other purposes, the log prints can be segregated from normal non-auditing traffic. If the logs were to be interleaved with day-to-day print jobs then the integrity of the printed copies could be called into question, or system users might remove crucial paper trail evidence from the environment.

9.4.2 Dangers of Optical Media

Having the logs written to optical media may mitigate some of the risks associated with modification, but this also raises the issue of trust with the personnel tasked with replacing and storing the CD media. If an IT administrator was to replace the disk with a falsified disc of records, then malicious or fraudulent activity could go undetected. Such an activity requires that appropriate procedures and processes surround the log media and disc rotation, and this is an excellent example of where the "Two Man Rule" would be well placed to require collusion between two or more members of staff to change the disc. This could be achieved by having a locked cabinet surrounding the log server, which required two keys held by two different members of staff. Both of these personnel would have to work in concert to open the secure cabinet, and this could mitigate the risk provided that procedures were well defined and staff had been properly trained. An additional paper based log would be useful as it would hold a record of access to the cabinet. Proper labelling (possibly using media stamped with tamper-evident hologram stickers) could also be applied to the media to ensure that valid media was being taken from the server before being stored.

9.4.3 Generation of Audit Trails

An important consideration for generating audit trails is making sure that the system is in fact generating the right logs, or even logging at all. A compromised application may have its logging disabled or modified by the attacker which will render the resulting logs inaccurate, so this raises the question about if auditing is embedded within application, the operating system, external to the system or a combination of all three. If an attacker is able to inject a root-kit into the system, then is the whole system and its logs trustworthy [32]? The generation of audit trails will of course create additional requirements for disk space, which is an important consideration. The question of the right data being logged is also key to ensuring

that appropriate response is taken to flagged issues. If every single transaction is logged, and every piece of information about that transaction, is it generating too much data to actually be analysed, and are the organisation's administrators actually looking at these logs? The use of intelligent filtering and alert thresholds may help in this respect, but defining logging levels and verbosity is itself a significant project which requires that the IT administrative staff are fully aware of the system's capabilities and have enough training and knowledge to implement the right level of logging for the right sorts of transactions and activities happening within the system environment and applications.

Chapter 10

Other Relevant Features

10.1 Redundant Array of Inexpensive Disks (RAID)

OpenBSD has driver support for a number of hardware RAID cards from a variety of vendors including Adaptec, 3WARE and Areca [59, RAID Support]. This allows system administrators to increase availability of systems in instances of a failed disk (provided that the RAID set has been properly configured). In addition to hardware support, OpenBSD also contains support for a software based RAID system called RAIDframe [63, page 172]. This system allows an administrator to define volume sets and configure RAID1 mirroring at the Operating System level. At the time of writing, RAIDframe is limited to support for non-booting partitions only, which means that the root / partition cannot be mirrored (this is from where the bsd kernel boots). Whilst partitions can be configured in a mirrored configuration to provide failover support for data, if the actual root partition becomes damaged then action will be required to boot the system as the root drive will be inaccessible. This limitation is owing to the fact that the support for RAIDframe lies within the kernel, not the hardware itself. This "chicken and egg" situation means that in order to access the kernel the system must be able to boot from a non-RAIDframe enabled root volume. This problem can be avoided by utilising hardware RAID, which acts transparently to the operating system (although drivers are still needed for the hardware RAID device once past the initial boot sequence). For systems where a bootable root partition and battery backed caching is not critical, RAIDframe can be a cheaper option for implementing disk mirroring.

In a financially sensitive environment, there is a need to ensure that data is available for processing when necessary, and the use of RAID can help to provide a degree of resilience to hard disk drives as data can be striped or mirrored across a number of physical disks, depending on the RAID level selected. Different RAID levels provide different levels of resilience

and performance, depending on which one is selected. For example, an FSE might choose RAID 1+0 (Mirrored Stripes) for servers requiring high performance, but this demands that twice the amount of physical storage is available (200 GB RAID 1 requires two 200 GB Disks).

10.2 Network Time Protocol (NTP)

It is important to note that properly synchronised times on systems are needed to help ensure accurate maintenance and running of scheduled tasks. Additionally, in the event of a security breach the production of logs with inaccurate times may be inadmissible in court or if allowed, heavily contested by defence counsel and used to weaken the prosecution's case. Such discrepancies would also be unacceptable on an audit report. In a Financially Sensitive Environment, the accuracy of time is vital to transactions, as many are time-dependent or based on duration (for example, fixed-term deposits and bonds).

One of the major issues associated with logging is the need to ensure accurate and well synchronised time [82]. Network Time Protocol (NTP) is an unsecured protocol, so the OpenBSD developers have created a sub-project called OpenNTPd [55], which provides a privilege separated time synchronisation daemon. Whilst OpenNTPd does not address the security of the network traffic (i.e. the integrity of the time data) owing to contraints of the NTP protocol and the need to maintain compatability with other NTP time servers, it does address the faults with the ISC NTP implementation by keeping the privileged parts of the network daemon separate from those that do not require privileges.

In recent years, work has continued to support accurate time sources in OpenBSD [1], including the creation of the mbg(4) driver to support a range of industrial-grade radio synchronised clocks manufactured by the German company, Meinberg Funkuhren. These devices allow time synchronisation between the global GPS system (accessed through nmea(4)) and also the German DCF77 time signal. Some organisations may require a high-quality and high-accuracy time signal such as that provided by radio clocks or GPS for their business applications.

Chapter 11

Hardening OpenBSD

OpenBSD prides itself on being "Secure by Default", and comes with minimal services and daemons running in its freshly installed configuration [58]. Other operating systems come with many services activated by default including server daemons such as web server services. Examples of this are Solaris 9 [27] and Windows 2000 [3]. A system administrator unaware of such default functionality may leave services running, exposing the server to potential routes of compromise through inadvertent attack vectors (for example, a dedicated database server would most likely not require a web server service to be started). Such default states require that an administrator 'lock down' the system by hardening the default configuration including disabling unnecessary services, reconfiguring needed services or altering basic system parameters. Whilst OpenBSD is delivered in a minimal state, there are some services still running which many systems administrators may wish to disable. In addition, certain configuration parameters for some services could be tightened up further by default, for instance the default configuration of the X Windows server listening on 6000/tcp.

11.1 sshd Configuration

The OpenSSH [60] server daemon which is included as part of the base installation of OpenBSD is configured via the `/etc/ssh/sshd_config` file [51]. Within this file the configuration parameters can be altered in a number of ways, from starting sshd listening on IPv4 only, to enforcing authentication by public key.

11.1.1 SSH v1 and Man-in-the-Middle

At the time of writing, the default parameters within `/etc/ssh/sshd_config` allow the use of both SSHv1 and SSHv2. SSHv1 is susceptible to a man-in-the-middle attack [62], but the merits of SSHv1 versus SSHv2 have been

debated, with Theo de Raadt even stating that he believes that the added complexity of SSHv2 may have additional security issues [9].

11.2 Disabling Unnecessary Services

As described above, OpenBSD installs by default with far fewer active services/daemons than other operating systems. However, there are still some smaller services enabled through the super daemon inted (configured via `/etc/inetd.conf`) which are frequently unused and can be removed - these include ident, daytime, time, comsat and chargen. These can be commented out in inetd.conf or preferably the inetd super daemon can be disabled by setting `INETD="NO"` in `/etc/rc.conf`. Fewer services listening on network ports will mean that there are fewer routes for potential exploitation.

11.3 Legal Warning banner

11.3.1 Background

In certain jurisdictions, a legal warning banner may be mandatory to alert users that their activities may be logged and monitored. Such banners can be configured in OpenBSD to appear on network login (via SSH or Telnet) and additionally through the "Message of the Day".

11.3.2 Message of the Day

Upon login to most UNIX systems, the user will be presented with a "Message of the Day" set by the system administrator to alert her to various issues or notifications of importance either to the system itself or for the wider organisation. In OpenBSD this file is found at `/etc/motd`. Such files are commonly used to communicate information such as planned maintenance on the system, and when it will be unavailable. Such a file can also be used to display security-related information, such as reminding users of the Acceptable Use Policy, or a secondary copy of the legal warning banner to ensure that users are made aware of organisational policies. In an FSE context, this file could also be used to remind users of the importance of data handling, or how to contact their data protection/information security officers should they detect suspicious activity.

11.3.3 Telnet and SSH Banners

As described above, Telnet and sshd banners can be configured to be displayed upon login. These can be symlinked to a single banner file to aid in administration through the use of `ln -s` and creating symbolic links to the desired standard text file. The banner for sshd is set in `/etc/ssh/sshd_config`.

11.4 Basic pf Firewall Rules

11.4.1 Restricting Administrative Access

Whilst the configuration of a firewall rule set is very much dependent on the required functionality of the system in question, a simple pf policy can be put in place to restrict access to the SSH port for remote administration. In the example below, 10.0.0.1 is used as an example adminstrative workstation.

```
# Macros, define network interface physical reference
int_if  = "dc0"

# Skip checks on local traffic
set skip on lo

# Scrub all inbound packets
scrub in all

# Default Deny Policy
block inet6
block in all
pass out

# activate spoofing protection for all interfaces
block in quick from urpf-failed

# Allow SSH from Admin station only
pass in quick on $int_if proto tcp from 10.0.0.1 to $int_if port
     ssh
```

11.4.2 Limiting Outbound Traffic

Depending on the system, there may also be a requirement to limit outbound traffic originating from the system (for example, to restrict network access to malicious code which may have been installed onto the system). In order to achieve this, the `pass out` clause can be removed and explicit permitted traffic rules defined to stipulate what outbound connections are acceptable based on organisational and/or system security policy.

11.5 Removing TCP Listener from X

OpenBSD's default configuration for X Windows allows remote users to connect to the X server over TCP port 6000. For many organisations they may wish for only the local user to be able to connect the X server, and such functionality would present another potential route for a remote attacker to gain access to the system. The default TCP port listener can be disabled

by modifying the file `/usr/X11R6/bin/startx`, changing the value of parameter `serverargs` to `"-nolisten tcp"`. The next time the X server is restarted, the X server will no longer be listening on 6000/tcp - this can be verified by running `netstat -an | grep 6000` - if nothing appears then the port is no longer listening, and remote connections inbound to the X server will no longer be possible. Such access could additionally be restricted by placing an appropriate pf firewall policy on the system and enabling the pf functionality to filter out such connections to the system.

Chapter 12

Systems Administration and OpenBSD

12.1 Patching OpenBSD

One of the key elements to administering any system is keeping the software up-to-date, especially patches for software with known defects or exploitable weaknesses.

12.1.1 Patching Systems - Release, Stable or Current?

OpenBSD can be kept up-to-date in a number of ways. The CD-ROMs released approximately every six months can be patched with the errata fixes from the OpenBSD website [42], whereby affected portions of code are patched and recompiled directly. Other options include following the -STABLE or -CURRENT branches of the OpenBSD CVS. STABLE contains the latest release, plus all errata fixes and fixed ports, whereas CURRENT is reflective of the current source tree which is under development. Every six months, the CURRENT tree is tagged as beta and made ready for the next release, pending testing. A 'snapshot' of the -CURRENT tree is regularly compiled and made available through the OpenBSD project's FTP servers.

Keeping systems up-to-date in this way may be fine for one or two systems, but what about organisations with many systems? Is one expected to manually re-patch every single system? Does a systems administrator want to keep a complete copy of the source code on each and every system, and make sure that it is kept fresh? Most likely the answer to this would be no, as the overhead of such activity would be significant and cause a time issue for systems administrators. A solution to this problem lies either in the 'Binary Patches' or Releases tools, which are described below.

12.1.2 Binary Patches

In recent years a non-OpenBSD project called binpatch [66] has sprung up to try to fill the gap of applying binary patches (much like those distributed for other operating systems such as Microsoft Windows and HP-UX) to OpenBSD systems. This project aims to give system administrators the ability to generate their own binary patches to apply to their systems, which removes the need to build internal releases through which to apply patches via an upgrade by installing a custom 'Release'. However, it is noted that the OpenBSD developers have made it very clear that this project is not official and not associated with the base system [67]. At the time of writing, no official binary patch tools are available or planned.

12.2 Making Releases

12.2.1 Overview

The OpenBSD system allows for the creation of "releases" [49], whereby an organisation's system administrator can take one copy of the source code, keep that one copy up-to-date with fixes, add in whatever organisation-specific changes are required, and blend in post-install scripts to make system deployment easier through automation. A security conscious systems administrator could then focus on one trusted code base, and increase her confidence that the systems being deployed within the organisation were being done so from a single point of control.

12.2.2 Single Code Base Security?

An organisation with a requirement for the "Two Man Rule" could configure the source code system itself to enforce higher levels of auditing, and require at least two members of staff to make a change to the code base. Such controls could be put in place by way of a source code database, whereby a change was input by one administrator, then flagged for review and approval by a second administrator. Such controls may help an FSE achieve compliance with ensuring proper oversight of changes to systems. The are currently no tools to achieve this with the basic OpenBSD base system, they must either be developed in-house or a third-party contracted to develop them as bespoke software.

12.3 Change Control

Whilst outside the scope of this project, it is worth mentioning that "change and release" control is vital to system stability, security and ensuring that

the enterprise platform remains compliant within the realms of the established corporate governance framework. Any changes to a system should be properly authorised, processed and documented. OpenBSD contains no built-in controls by default to this end, however it is possible to extend functionality (as with almost any open-source project) by adding additional code or implementing a separate package to handle such matters. Proper change management, authorisation and management oversight of such activities are often core elements of standards and regulatory/statutory practices, including Sarbanes-Oxley (if one were not to, could management prove 'adequate control'?), ITIL and ISO/IEC 17799. More recent standards such as the Payment Card Industry Data Security Standard (PCI-DSS) explicitly sets out the need for proper change management [64], and this is also a key part of best-practice methodologies like ITIL and COBIT [30].

Chapter 13

Criticism of OpenBSD

13.1 IPv6 mbuf Patch Issue

The OpenBSD team's response to a security issue in early 2007 came under fire when Core Security discovered a problem in the way the kernel would handle a fragmented, customised packet [6]. Despite Core Security's concerns about the severity of the flaw, the OpenBSD team initially responded with a patch marked as a "Reliability Fix", arguing that the flaw could not be exploited so far down the mbuf chain. Core Security responded by developing Proof of Concept (PoC) code to exploit the vulnerability, and the OpenBSD team then reissued the patch and labelled the new version as a "Security Fix". The code flaw, and resulting drama by the developers, was covered by several major technology news sites, including Slashdot.org [72].

13.2 Securelevels

In 2005, OpenBSD founder Theo de Raadt received criticism for his response regarding a flaw found by RedTeam Pentesting in "securelevels" (a common component of the BSD and Linux families) from the high-profile information security website SecurityFocus [70]. Securelevels is a function within the kernel which seeks to prevent certain behaviour even from the root user. An example of this is the changing of extremely sensitive files. In this particular flaw, an attacker could mount a file system over the top of immutable files, effectively linking an attackers choice of hostile code to previously trusted binaries. Theo de Raadt's response was:

> *"Sorry, we are going to change nothing. Securelevels are useless."*

The SecurityFocus article went on to explore the impact of such a statement, and questioned why, if securelevels are useless, are they still kept within OpenBSD? The argument made by SecurityFocus was that if in fact

this code is “useless” why not remove it and thereby not give users a false sense of security. Further discussion on the misc@openbsd.org mailing list discussed removing the code from OpenBSD and using more advanced technologies such as Systrace instead. To this day, the securelevels functionality remains within the operating system. The article also states that NetBSD is immune to this attack, but this is owing to the fact that NetBSD does not allow any mounting by default [68].

Chapter 14

Infrastructure Diversity and Security

One of the core considerations for any security implementation is evaluating the quality of products intended for deployment. If a single technology is chosen to provide security across multiple layers of the architecture, then a failing in that product may affect all layers on which it is installed. For example, in highly sensitive environments employing two or more firewalls in serial to filter traffic, if the same firewall product (and by extension, version and configuration) is used, then a flaw found in the outer most perimeter would also be present on the inner perimeter, making an attacker's job a case of simply exploiting the same flaw twice to gain the required access.

By diversifying the product range used in such scenarios, a flaw in one product may not be present in others, requiring an external attacker to penetrate two layers of firewall defence before being able to breach the inner perimeter. A diverse environment brings with it other challenges and considerations, notably the need for support staff to be trained in both product lines, patch and maintain two sets of systems and also maintain credentials for both systems unless a single sign-on is supported - indeed this too may lead to issues where one compromised set of credentials could then be used to alter security devices to the attacker's desired configuration.

An organisation must perform an adequate risk assessment to explore such issues and determine their appetite for risk and available budget. Wider organisational issues may also drive a particular technology's usage, and all developments of hardware and software should be carefully considered.

Part III

Compliance, Financial Regulation and Control

Chapter 15

Overview

A challenge for business (and in particular, financials firms) is ensuring that organisational policies, procedures, processes, guidelines and standards are aligned not only with international regulatory standards (e.g. Basel II) but also national law (e.g. US Sarbanes-Oxley Act [80]) and local regulators (United Kingdom Financial Services Authority [18]). Coupled with the ISO standards for banking and financial systems (e.g. ISO/TR 17944 [29]), standards for quality (ISO 9001) and information security (ISO 17799 [28], 27001) plus best practice (COBIT [30]), bringing all of these together and synchronising them with the needs of the business can be an exceptionally difficult task, especially in the context of large organisations with heterogeneous platforms and a diverse array of legacy systems.

With the introduction of new compliance standards, such as those developed by the credit card industry (Payment Card Industry Data Security Standards (PCI-DSS)) often comes prescriptive and very specific control sets [64] which must be adhered to in order to be certified as an authorised partner in financial networks. PCI-DSS is an excellent example of specific controls leading to accreditation, as a registered QSA (Qualified Security Assessor) who is trained and recognised by the PCI partnership must be engaged to provide certification for certain organisations, depending on their transactional and monetary volumes within a financial year.

This section discusses some of the applicable standards and regulatory/statutory areas which could affect a Financially Sensitive Environment, and how OpenBSD and its security features might be used to create and maintain compliance/certification.

Chapter 16

Aligning Governance, Oversight and Regulation

With the vast array of different legislation, regulatory bodies requirements and various standards, organisations have to find a way to consolidate and understand their obligations, and map these to appropriate security controls which are both cost-effective and sufficient to ensure compliance. Some areas of major controls packages and standards can be mapped to one another fairly well, but others may require much more careful alignment to ensure that the nuances and specifics are not lost in the mapping process. Having one consolidated list of controls may aid an organisation in respect to its security strategy and vision, whereas multiple lists may confuse issues and result in non-compliance where certain controls have been missed owing to misalignment or a failure to understand the requirements. Examples of some of the areas which are commonly encountered include:

- Sarbanes-Oxley Act 2002 (USA)
- Financial Services Authority (UK)
- Basel II (Worldwide)
- Payment Card Industry Data Security Standards (PCI-DSS)
- ISO/IEC 17799 (leading to ISO 27001 certification)
- ISACA COBIT 4.1

Chapter 17

Mapping Requirements to OpenBSD

The table below is a sample of a proposed mapping model for various different control sets and regulations (such as ISO/IEC 17799:2005 [28], COBIT 4.0 [30], PCI-DSS v1.1 [64]) against OpenBSD's security features:

COBIT 4.0	ISO 17799:2005	PCI-DSS 1.1	OpenBSD Control(s)
DS5.10 Network Security.	11.4 Network Access Control	Requirement 1: Install and maintain a firewall configuration to protect cardholder data.	pf firewall
DS5.5 Testing, Surveillance and Monitoring.	10.10 Monitoring	Requirement 10: Track and monitor all access to network resources and cardholder data.	Syslog, system accounting
DS5.9 Malicious Software Prevention, Detection and Correction.	10.4 Protection against malicious and mobile code.	Requirement 5: Use and regularly update anti-virus software.	Systrace, clamav (3rd party), mtree

In the table above, we can see that the mapping of network security/access control could be achieved by utilising the pf firewall (discussed in Chapter 5) with OpenBSD. The requirements for auditing could well be met by proper configuration of Syslog in OpenBSD, provided that there are adequate controls over the resulting log files (as we saw in Chapter 9). Other requirements and controls can be met by many of the other security features discussed in "Part II - OpenBSD: Technologies and Features", and indeed some of the controls are discussed in the context of defining a secure environment for a real-world Treasury department in "Part IV - Case Study:

Securing Treasury".

For different environments, an organisation may define different weightings in order to give a preferred balance, based on the appetite for risk which is accepted by management within the organisation. For heterogeneous environments, it could be that certain controls do not map accurately to a technical solution, so compensating or mitigating controls may be put in place by cross-referencing to an alternative suitable control, or possibly utilising a different platform to provide the necessary control within the environment (for example, by placing a capable application proxy in front of a problematic application server which is unable to properly support a suitable security control). Proper application of controls within appropriate areas of the business are vital to ensuring the security of the whole environment, especially for financial environments which may be exposed to fraudulent activity by internal staff [14, page 151].

Chapter 18

Compliance Challenges

18.1 Sarbanes-Oxley Act: 'Adequate Control'?

The United States of America's Sarbanes-Oxley Act of 2002 [80] (also known as the Public Company Accounting Reform and Investment Act) has imposed a number of obligations upon many firms in the United States. One of the parts of the law often used within an IT context is Section 404. This section requires that management establish and maintain *"adequate internal control structure and procedures for financial reporting"* and that duly appointed external auditors assess the effectiveness of such controls and procedures [80, page 45].

However, it is worth noting that nowhere in the entire text of the Act are there any specific requirements of 'Information Security', so this raises the question as to what constitutes an 'adequate control'. To this end, many information technology and security firms have been eager to pursue new lines of business and have actively marketed their product lines to organisations seeking to ensure compliance with the legislation, much of it under the mistaken notion that compliance can be achieved with a single product [31].

One approach to looking at adequate control might be to start with an understanding of the environment that is reflective of the organisation's risks (discovered by proper risk assessment), and then utilise best practice and standards (e.g. ISO 17799) mapped to statutory/regulatory requirements (e.g. PCI-DSS, HIPPA) to generate a security baseline from which the organisation can build the necessary wrap-around policies and procedures to ensure the quality of system configuration, application design, user platform usage and ongoing administrative maintenance. It could be argued that starting with a recognised secure approach to systems design, such as that by the OpenBSD developers, is one of the first steps to defining a secure

baseline on to which ‘adequate controls’ may be built.

18.2 Approaching Risk

Selecting a secure operating system, defining a suitable configuration and deploying it within the enterprise should be seen as an action arising out of a structured, planned and reasoned security assessment where failings justify such actions. Mandating that all systems are configured to the highest possible standards, with extreme measures over change/release control, management sign-off authority and least privilege enforcement may at a high-level seem like the answer to security issues, but without a pragmatic approach to assessing and managing risk, an organisation may spend more money than is necessary, or even apply security controls to the wrong areas.

Chapter 19

Open Source Software in Business

19.1 Acceptance and Adoption

As with licensing, the rate and level of acceptance of open source software is outside the scope of this project, but it is worth noting that in recent years such solutions have been given a better opportunity to find their way into organisations' technology platforms. An example of this is the policy of considering open source solutions issued by the UK Government - a policy document issued by the Cabinet Office laid down instructions for Open Source solutions to be given review where possible [5]. This driver towards including Open Source solutions in evaluation against traditional commercial closed-source solutions was one of the first steps in the UK to look at such free software at a national level.

19.2 The Costs of Implementing Open Source Software

Whilst free, open-source software such as OpenBSD may be free from licensing costs, business still must pay for the hardware upon which such software runs and the implementation and configuration costs for skilled personnel to install and customise such software to achieve the necessary business functions. Such costs are always unavoidable, as businesses must pay their staff, purchase hardware upon which to run software and power the equipment with non-free utilities such as electricity. As with any new software/hardware project, there are also associated integration costs for staff to ensure that new platforms can interoperate with existing enterprise systems. There is also a strong likelihood that training will be required to educate staff on the use of new systems, and this too is a cost which must be borne by an

organisation.

A challenge for organisations is balancing the costs of implementing any system against the business benefit, and it could well be that the cost of training staff in a new operating platform may be in excess of paying the licensing fees for commercial software with which they are already familiar. It is up to management to ensure that an organisation's information systems strategy is aligned with the needs of the business, regulatory and statutory obligations, and the security requirements necessary to operate effectively. Open source software is not a magic tool through which organisations can shed all their IT costs, but can be considered as an alternative to closed-source, commercial solutions. Depending on an organisation's appetite for risks, all factors must be balanced.

19.3 Continuity and Source Escrow

Organisations may find that open-source software delivers an additional assurance by making the original application code available, giving the organisation the freedom to hire their own programmers to perform maintenance and functional extensions. Having access to the source code also removes the necessity to arrange source escrow for applications (whereby proprietary third-party code is held by a trusted entity in case of application vendor bankruptcy, for example). Having the source code to software may also give an organisation opportunities to quickly extend application and platform functionality to meet new business objectives and respond to client needs. However, this must also be balanced with the need to hire and/or train personnel to understand and extend the source code, and the costs of doing this may outweigh the costs of having a third-party maintain code, especially for applications with a low level of change. Contractors could be brought in to maintain low-change code bases, but this also incurs costs and implications.

Financially Sensitive Environments may find that having the source code to OpenBSD allows them to extend security functionality with organisational or departmental specific controls which better support their business processes and align with existing audit procedures.

Part IV

Case Study: Securing Treasury

Chapter 20

Background

20.1 Confidentiality and Context

For the purposes of commercial confidentiality, the following security assessment has been sanitised of potentially identifying material including, but not limited to, hostnames, IP addresses and office locations. For brevity and clarity in keeping with the scope of this dissertation, sections pertaining to physical security controls and treasury processes have been omitted. In order to aid context for presentation of this assessment within this project, additional sections have been added to explain the Treasury department integration within the wider organisational network structure and the challenges it faces.

20.2 Overview

This case study is being included owing to the specific context of security within a financially sensitive environment, in this instance a Treasury department. The assessment demonstrates areas which may be problematic in such environments, and acts as a foundation for introducing real world examples of where OpenBSD (and the security controls described in "Part II - OpenBSD: Technologies and Features") may be employed to address such issues within the wider scope of managing information security for these types of business unit. This case study will also show how some of issues described in "Part III - Compliance, Financial Regulation and Control" map to a real-world FSE and the potential security benefits afforded by placement of OpenBSD into such an environment.

Conducted by the author during 2006, this assessment was undertaken whilst acting as an information security consultant to a large international financials firm. The scope of the security assessment was the electronic, physical and process security of the Treasury department based in Berkshire.

20.3 Security Assessment Drivers

The driver for conducting this security assessment arose from the Treasury Manager's concern that the technology deployed within her department was dated and that the previously assigned dedicated technician had been made redundant by senior management without her consent or knowledge, leading to a lack of maintenance in the environment. In addition, the upcoming rollout of Windows XP Service Pack 2 across the rest of the enterprise had not taken into consideration the enhanced security needs of the Treasury department as neither the Treasury Manager nor any of her subordinates had been consulted about their business requirements or concerns.

The question about security was raised at a quarterly review of the EMEA auditing output and passed to the author whilst still under contract with the organisation. Endorsed and authorised by the Chief Information Security Officer, the security review was conducted with full knowledge of all Treasury staff.

20.4 Outsourcing, Offshoring and Risk

During summer 2005, the organisation had decided that certain Treasury functions would be off-shored to a team in Budapest. With the business justification that the skills were available at a lower per-head cost than was available within the United Kingdom, access to the main accounting systems were to be gradually opened up to the Budapest team over a transition period of 12-18 months. The Chief Information Security Officer advised the author that the Treasury security profile had to accomodate the changes in business operations and that subsequent recommendations and designs must reflect the new working practice.

20.5 Treasury within the Wider Infrastructure

20.5.1 Segregation

The Treasury department was originally intended to be a secure compartmentalised section of the wider core network within the organisation, with a firewall and strict policy to restrict network traffic between the "Treasury Secure LAN" and other networks.

20.5.2 High Security?

In addition, the historical view of the Treasury department by both non-technical Treasury staff and technical IT support teams was that the systems were afforded a much greater degree of security than the rest of the corporate

platform. As came to be discovered, this was not the case, and no documentation or evidence could ever be produced by the technical teams to support the original claims of hightened security controls ever having been put in place on any Treasury server, workstation or network device.

20.6 Conducting the Security Assessment

After written authorisation was received from the Treasury Manager and Chief Information Security Officer, and using the organisations information security policies and standards documents, the security assessment was conducted on-site at Treasury over ten working days, and included:

- Interviews with staff, including the Treasury Manager, IT Liaison and Chief Cash Controller
- Network Security Assessment: Scan of "Secure Treasury LAN"
- Console-based investigation of system configuration
- Review of manual processes and procedures to establish existing practice and baseline context for electronic controls

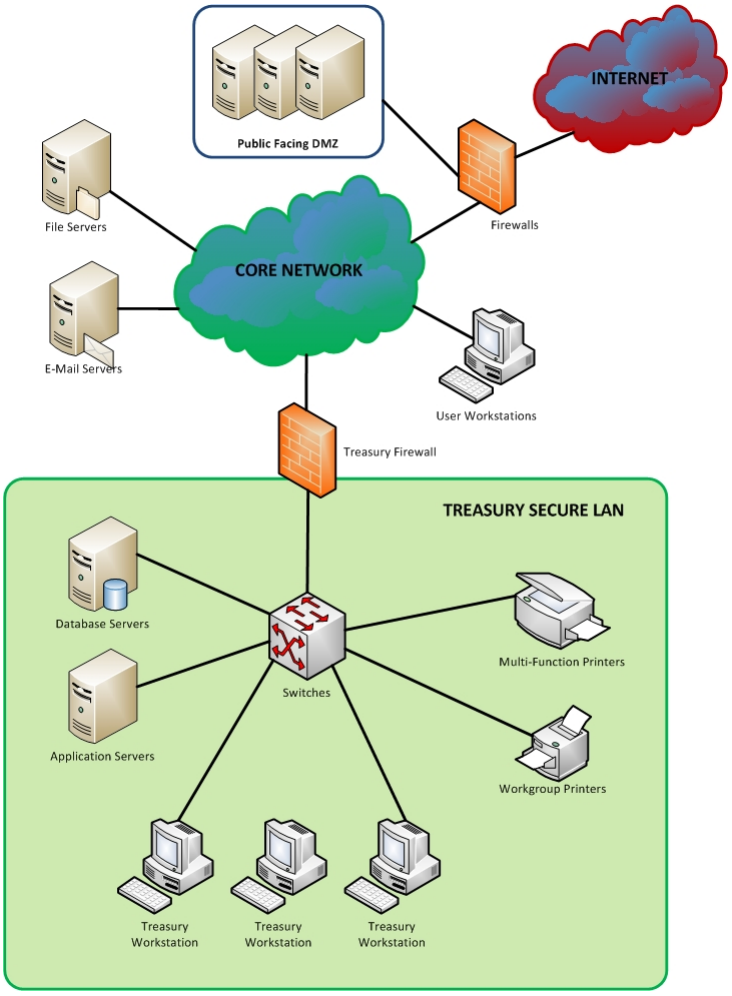
An assessment was then produced, documenting issues identified and to provide recommendations on remedial actions, advising short to medium term compensating controls where appropriate in instances of significant design issues.

Chapter 21

Findings

21.1 Treasury Local Area Network

All internal company networks utilise private addressing as described in [22]. The Treasury network has the dedicated network range of 10.252.16.0 with a subnet mask of 255.255.255.0 (a /24) allowing for a total of 254 usable IP addresses. The department consists of 35 workstation systems and 5 dedicated servers, with servers being allocated static IP addresses and workstations collecting a lease for an IP address via DHCP (Dynamic Host Configuration Protocol) [23]. The Checkpoint firewall gateway system was assigned IP address 10.252.16.254, with all systems set to use that as their default route to other networks.



21.2 Server Configuration

All servers within the Treasury Secure LAN are running either Windows NT4 Server or Windows Server 2000. No hardening has been performed on these systems, and no system has had any vendor patches applied.

The detailed lists of Microsoft Windows services, file shares and enumerated accounts discovered have been omitted for brevity, clarity and context.

One of the major problems identified in the Treasury Secure LAN was the inappropriate configuration of the electronic payments system which feeds the BACS clearing processor. This system takes an electronic transaction file and sends it via a dedicated modem link to one of the payment banks used by the organisation. This server collects payment information from a variety of sources (including the Oracle Financials Server) and stores pay-

ment instructions (including beneficiary names, bank account details, BIC codes, currency and amounts). This plain text file is unencrypted, not digitally signed and is located in a world readable and world writable file share accessible to any system that can connect. Owing to the lack of any firewall rules, any system within the organisations core network could connect to this file share. The lack of any user or group permissions on this file and share means that any person can modify the file, and no fraud would be detected until at least 48 hours later when the bank reconciliation system would download the statement files for cross-checking against the authorised payments database. In addition, the system had no auditing enabled, meaning that any modifications to the file would be anonymous and almost impossible to track back to the origination point. It is worth noting that the system is scheduled for replacement in the future, but neither project plan nor timescales had been defined for the replacement at the time of the security assessment.

21.3 User Authentication

All users within the organisation have their user accounts stored within a Windows Active Directory, but the security assessment identified that the current Human Resources policy and lack of systems administrators meant that users who had left the company (for whatever reason) still had active accounts within the central directory for anywhere up to 12 months from the date of their leaving. In addition, no controls were in place other than quarterly audit checks, and even these often failed to examine the entire directory structure. At the time of writing, there are 47 Domain Administrators within the organisations domain, with sixteen of these accounts utilising generic usernames (such as ADMIN1), each with passwords commonly known to many of the IT administration team. This lack of accountability could cause a critical failure in security measures and exposes the entire network to unauthorised modification and possible fraud.

21.4 Workstation Configuration

All workstations within the Treasury Secure LAN are running Windows NT4 Workstation with Service Pack 6a. No hardening has been performed on these systems, and no system has had any post-SP6a patches applied.

The detailed lists of Microsoft Windows services, file shares and enumerated accounts discovered have been omitted for brevity, clarity and context.

21.5 Time Synchronisation

All servers and user workstations within the Treasury Secure LAN displayed incorrect times. No systems have been configured with NTP or Windows Time Service to collect the correct time from an authorised time source. In some cases the time on some systems was off by over 4 hours.

21.6 Outdated Operating System Software

Each user workstation was running an outdated version of an operating system, in most cases this was Windows NT4 Workstation, which is no longer supported by Microsoft. This means that security fixes and critical patches are no longer issued for this software, resulting in vulnerable systems which can be exploited by attackers. All servers are also running either Microsoft Windows NT4 Server or Microsoft Windows Server 2000.

21.7 Firewall

The Network team have advised that the firewall currently deployed to segregate the Treasury Secure LAN is Checkpoint 4.1. Despite having been deployed over four years prior, the firewall had not been configured, and the filter rules were set to allow any traffic to any destination on any port, resulting in a default permit and allowing any network traffic back and forth between the core network and the Treasury Secure LAN. As there is no traffic filtering or logging in place at all, the firewall itself is serving no security purpose.

21.8 Internet Downloads

As with other departments within the organisation, the Treasury Department is subject to web filtering to remove potentially hazardous downloads such as known viruses. Web traffic is filtered by Bluecoat Web Proxy Systems, located within the organisation's core network. However, the current settings permit users to download executable content and other binary files, which could contain unknown malicious software such as Trojan Horses or spyware, potentially affecting the confidentiality, integrity and availability of all systems.

21.9 Modems

A previous report filed by the XP Upgrade Rollout Team and the author's own on-site checks reveal that there are several systems in use within Treasury which utilise Modems to connect to external providers and systems.

Such devices are a major risk to corporate infrastructure as they bypass network filter controls usually enforced by the boundary firewall device. Whilst this security review did not check the individual security settings of each modem, it is noted from the XP Rollout Team's inventory documentation that in most circumstances the modems were configured with factory defaults, and all modems permitted unauthenticated inbound calls to systems, which could allow unauthorised persons to gain access to the systems to which they were attached, and if those systems were connected to the organisation's network, function as a beach-head from which additional attacks could be generated.

21.10 Network Access Control

The Treasury Secure LAN has no additional controls to restrict unauthorised systems from using its dedicated network. This means that any person can connect a system (e.g. a rogue laptop) to the LAN and obtain an IP Address via DHCP lease. Such a system can then freely communicate with all workstations and hosts on that subnet. The lack of physical security controls within the Treasury environment means that rogue systems could be connected, and used to directly attack Treasury and the rest of the core network - including the payment processing systems.

21.11 DNS Configuration

21.11.1 Overview

The following details pertain to the DNS servers which serve the whole internal corporate network. All Treasury systems (workstations, servers and network devices) utilise these DNS servers.

21.11.2 DNS Server 1 (Primary)

FQDN	ukdns02.uk.example.com (PTR record for 168.76.149.8)
Hostname	Ukdns02
Telnet Active	Yes
Telnet Banner	None, but version revealed as HP-UX B.11.11
SSH Active	Yes
SSH Banner	None
Version.Bind	8.1.2 (reported*)

21.11.3 DNS Server 2 (Secondary)

FQDN	ukdns03.uk.example.com (PTR record for 168.76.150.20)
Hostname	ukdns03
Telnet Active	Yes
Telnet Banner	Yes, also version revealed as HP-UX B.11.00
SSH Active	Yes
SSH Banner	None
Version.Bind	9.2.0 (reported*)

21.11.4 Issues

The outsourced support team was consulted and the INITPRO UNIX Administrator produced a document from November 2005 describing how the two servers above were to be decommissioned in favour of the implementation of the new QIP IP and DNS Management system (administered by the US Network Team). The two legacy DNS boxes were to be left running until the QIP systems had stabilised and the pointers had been updated to reflect the new DNS server addresses. These pointers still need to be updated in the DHCP Lease settings on whatever QIP or Wintel box is running the DHCP services as these old legacy DNS server addresses are still being returned by the DHCP lease assignment to clients.

These two critical DNS servers are running on the internal core with telnet access still enabled, plus all but one of the services are also not displaying an appropriate legal warning banner on connection. There are also multiple known Denial-of-Service and Unauthorised Code Execution vulnerabilities in the BIND versions reported. If these two DNS servers were to be disabled by poor configuration, exploit or denial-of-service attack then it would disable DNS resolution for all staff workstations and other servers that rely on them.

A more in-depth port scan of these DNS servers has not been conducted as there is a high possibility of disrupting service. The issues discovered with DNS do not affect just the Treasury department - as these servers are located within the core network and used as the primary and secondary, every single system which queries DNS via these two systems is potentially open to flaws with either of these servers. An attacker who compromised one or both of the DNS systems could modify the DNS server configuration and “posion” the DNS database to redirect traffic to systems of his choosing, or alternatively wipe out the DNS entries to perform an effective Denial of Service on all systems which rely upon DNS.

```
* Reported Version Number discovered by:
```

```
C:\nslookup
> set type=txt
> set class=chaos
> version.bind serveraddress
```

Chapter 22

Recommendations

Below are some of the recommendations made to the organisation's management regarding remediation required in the Treasury department. Owing to the poor levels of security present, it was recommended that the organisation swiftly begin planning and design to work towards a fresh security approach for Treasury, as can be seen in Chapter 23.

22.1 Network and Firewall

As stated in Section 20.5.2, personnel were under the misguided belief that the Treasury network had been segregated from the rest of the core network by a firewall with a restrictive policy to limit the systems and protocols which could be used. It is recommended that this actually be put into place by updating the firewall to the latest version, verify that the equipment is registered an as asset under the control of the network team (and supported by the vendor as per organisational policy) and then configure a rule set which accurately reflects the needs of the Treasury. Preferably the single Checkpoint firewall should be replaced with a pair of OpenBSD systems running pf and CARP (as described in section 5.2.5) which will provide resilience in the event that a single system should fail - these should be configured in a full mesh from the Treasury switches which will be provide continuity in the event of the failure of one firewall, cable, switch or associated power supply. The configuration of firewalls is not only supported by the organisations information security policy, but is also recommended in "Section 11.4.5" of ISO 17799:2005 [28] and "Requirement 1" in PCI-DSS [64]. To address the staff ability to download potentially malicious binaries, a web proxy system could be developed to vet all web browsing content, and prevent binaries from being downloaded. Such a solution could be the use of an OpenBSD server running the Squid Web Proxy. Alternatively, the existing Bluecoat Web Proxy policy could be updated to restrict binary downloads, but this would then affect the entire organisation and may have impact on areas of

the business which require the ability to perform such downloads.

22.2 System Configuration and User Authentication

It is recommended that existing HP-UX systems be replaced with OpenBSD servers running the business applications in emulated mode. This must first start with a secure baseline (Hardened as described in Chapter 11 of this document, and in line with the Global UNIX Security Standards), change management process and oversight programme approved by the Global Information Security Group. The business applications hosted on OpenBSD servers will then benefit from Memory Protection Features (as described in Chapter 8). In addition, each application should be assessed to ensure that it is run in a restricted user context (i.e. not as root) and where possible, constrained with Systrace (providing that a solution is found for the major security flaw described in Section 7.3 of this project). Each server should also be enabled with the pf firewall to limit administrative access to approved management workstations and also limit client access to only those IP addresses with the Treasury Secure LAN - this will limit staff access to the system. User credentials should be audited and linked with the core Directory Services, possibly using Lightweight Directory Access Protocol (LDAP) extensions available for OpenBSD.

22.3 DNS

As described in 21.11, the issues with the DNS are not Treasury specific. It is vital that these systems are patched (to remove the potential for exploiting well-known flaws) and hardened against attack. At an absolute minimum, Telnet must be disabled and only SSH used with public-key authentication, as stipulated in the organisations UNIX security standards document. The standard organisational legal warning banner must also be applied to the SSH service configuration file. These two HP-UX DNS servers should be replaced with two OpenBSD-based systems running the hardened version of BIND which is present in the base system. In addition, pf firewall should be configured on each of these systems to only allow DNS requests from the approved core network, and rules can be put in place to limit connections to the SSH port to the approved IT administrative workstations. This will limit the ability of attackers to connect over anything except DNS, requiring the discovery of a flaw in the OpenBSD BIND implementation, which also runs as non-root restricted user. For additional security, the BIND service could be run caged by systrace, although as discussed in 7.3 there are known problems with systrace.

Chapter 23

Designing a Secure Future Treasury

23.1 The Challenge

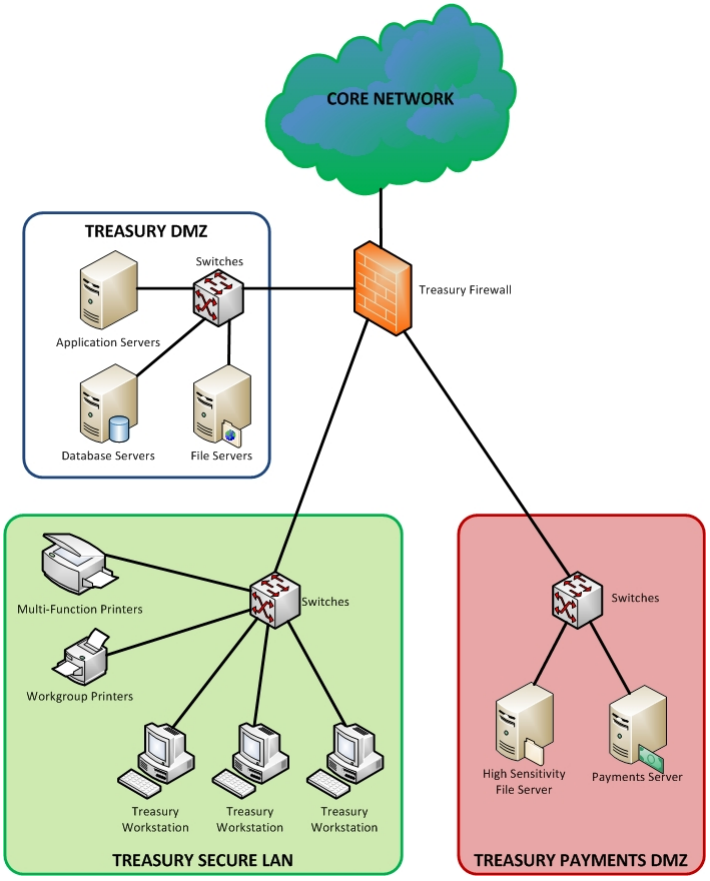
Tactical remediation of the issues discovered by the security assessment is the first step, but the question of its future security still must be answered.

23.2 Treasury Security Strategy

In the wake of the publication of the Treasury security assessment, the organisation's Global Information Security Group decided that a complete overhaul of the way that security policy was managed and disseminated was required. In addition, the author was requested to create a new secure network for Treasury, addressing the issues identified in the assessment and also providing a platform to ensure future security and adaptability for the Treasury department.

23.3 Technical Design

23.3.1 Network Diagram



23.3.2 Specification

The proposed future design for Treasury creates three separate network zones. Segregation of these zones is by use of a stateful packet filtering firewall (such as OpenBSD's pf) to control traffic between each Treasury zone, and between Treasury zones and the wider corporate network and external networks such as the Internet. These zones can be created either on separate physical switches, or by utilising VLAN technology on a single larger switch whereby switching blades and their ports can be tagged for each zone. The three network zones are physically cabled to firewalls, providing no direct physical link between any zones apart from that mediated by the firewall.

Treasury Secure LAN	The zone in which Treasury users and their peripheral devices are located. Minimal access to this zone inbound from other Treasury zones or any other network. No servers located here.
Treasury DMZ	A new semi-public zone housing standard applications, file servers and other server -side functionality.
Treasury Payments DMZ	A new Treasury-only zone housing the high-sensitivity systems such as the Electronic Funds Transfer (EFT) systems, and critical file servers for processing financial data. This zone can only be access by systems located within the Treasury Secure LAN - traffic from the Treasury DMZ, core network or external networks will be permitted to enter this zone.

A new baseline should be defined for a secure operating platform. Standard applications should be hosted on a server within either the Treasury DMZ or Treasury Payments DMZ as appropriate, with client workstations in the Treasury Secure LAN accessing necessary applications over thin-client connections. This will remove the necessity to have (and maintain) full client workstations in the Treasury Secure LAN, and result in minimal numbers of ports having to be opened between zones. A technology such as Citrix could be used to publish a working desktop to users, with separate servers publishing applications to the published desktop. A stripped-down desktop environment that only gives the users the applications they need would reduce the risks associated with having to maintain other non-essential software, and read-only client terminals would minimise disruption from potentially malicious code. Published desktop servers can be hardened against attack by removing extraneous software and drivers, and corporate branding and settings can be placed at a single point of enforcement.

Infrastructure Server - OpenBSD system running BIND (DNS) to act as a hardened DNS forwarder out to the wider core networks DNS server and dhcpd (DHCP) to issue IP Address leases to systems within the Treasury - this keeps infrastructure traffic from leaving the Treasury Secure LAN. The system should be configured with a pf firewall ruleset to prohibit any communication with the server other than DNS and DHCP requests, plus administrative traffic from the approved IT administration subnet.

Firewall System - As described in Section 22.1, a pair of rack mounted servers running OpenBSD to be locked in the Treasury's dedicated server

room. These servers to be running a minimal configuration of OpenBSD, with pf and CARP to provide high availability firewall functionality. Rule set to be configured to split out the various Treasury zones, with a port on each firewall trunked to the appropriate VLAN or physical switch (as budget allows).

A hardened log server (OpenBSD with syslog-ng/SSL) should be deployed in the audit zone (with the core network) to receive logs from the Treasury Zones. Any malicious intruder within the Treasury boundary of trust would then have to compromise the physically separate (and secure) audit zone in order to modify or delete logs of activities conducted within the Treasury Zones. The syslog consolidation server will act as a central point of collection for crucial audit data (as described in Chapter 9). The use of Secure Sockets Layer (SSL) to secure the syslog traffic addresses the lack of security in syslog messages broadcast over UDP. This is an example of where a third-party piece of software should be deployed, as OpenBSD's syslog implementation does not have SSL support (although some unofficial patches exist to add this feature).

In addition, IPSec can be used between servers to provide mutual authentication and also confidentiality and integrity to network traffic if ESP+AH is used. In this respect, a malicious entity attaching a network sniffer to the Zones (or modifying a network switch to re-direct all traffic to diagnostics/mirror ports) would not be able to read or modify the network traffic. The IPSec functionality could also be utilised to provide a secure tunnel between the Treasury firewalls to the new Treasury team in Budapest (as discussed in Section 20.4), which will create an encrypted channel between the Budapest office and the Treasury site in Berkshire, UK. This means that a malicious entity within the core network could not gain unauthorised access to the sensitive content and/or modify it in transit. This could be deployed either with another OpenBSD firewall in Budapest (creating a tunnel between the two firewalls running IPSec) or alternatively a transport between each Budapest Treasury workstation and the Berkshire Treasury Firewall, effectively providing the Budapest staff with a Virtual Private Network (VPN) between Budapest staff and the UK Treasury.

All servers should be configured with host-based intrusion detection tools (such as Tripwire, AIDE or mtree), regularly invoked by cron job or by daemon timer to monitor changes to critical system binaries and configuration files. Alerts relating to such changes should be directed to the organisation's information security e-mail address as a point of collection, so that the alerts can be actioned by an appropriate member of staff and the incident response procedures invoked. Such intrusion detection should detect if an attacker had changed any system binaries (such as inserting a root kit) or added

additional users with which to access the system, for example.

Provided that the Systrace vulnerability described in 7.3 can be rectified, each server should be configured to constrain the business applications present on the servers so that not only do they run as a restricted user ID, but are also limited by policy as to which resources the application may access and in what mode (e.g. file and network access).

OpenSSH (as described in Chapter 6) should be used on a dedicated file server sited within the Treasury DMZ to act as a file transfer system, removing the need for plain-text FTP exchanges in or out of the Treasury environment. The use of the sftp-server subsystem can be bound to specific users on the system to only permit such file transfers. This removes the in-transit risks to payment data files being sent into Treasury from other global branches' AS/400 and IBM Mainframe systems. In addition, all systems administration of UNIX systems (including OpenBSD) should be conducted via SSH connections, and no Telnet daemons should be enabled on any system.

Part V

Conclusions and Looking to the Future

Chapter 24

Future Challenges in Compliance

As new legislation and standards are introduced, there will be a growing need to find effective ways of mapping those requirements to organisational technology platforms. The business must not be hampered by the limitations of technologies (or if they are, there must be sufficient tools to implement compensating/mitigating controls) and the security, audit and compliance functions within the enterprise must be able to understand the impact of placing suitable controls in place, not only from a costs perspective but also in aligning the needs of the business with overall organisational goals and external obligations.

In Chapter 17, a basic mapping model was introduced to show how various controls from different standards and regulations could be aligned with one another, and suitable functionality from OpenBSD brought in to meet the needs of those areas. It is worth reiterating the issue of security diversity, explored in Chapter 14, and business leaders must have the right information available to them as part of appropriate risk management to ensure that security challenges are being met with the right type of technology, with the most suitable configuration balanced to the budget and administrative overheads which are acceptable. Multiple layers of security might be the best option for a business, but what populates those layers should be a considered and reasoned solution.

Chapter 25

OpenBSD's Roadmap

Owing to the volunteer nature of open-source community software development projects (such as OpenBSD) there is no real roadmap for development. Developers work on the project in their spare time, and as such will be motivated to add features which interest them or are relevant to applications which require certain controls to be put in place. The lack of such a roadmap makes each release of OpenBSD interesting, as although one can follow the CVS change logs or mailing lists, often many of the new features are only very clear once various pieces of new code have been stitched together and documented for release.

OpenBSD's desire to be the optimal system for security features is now being challenged by other operating systems, such as Sun Microsystems' Solaris. Solaris is now freely available and contains many of the security features previously reserved for Government clients (such as the Trusted Solaris Extensions) [75]. Solaris has true Role Based Access Controls (RBAC), something which OpenBSD currently lacks. With new developers joining the OpenBSD team (and the Operating System's profile being raised) there is a strong possibility that additional functionality such as RBAC could be added in the future. Such features could strengthen OpenBSD's commercial acceptance, and see its usage in Financially Sensitive Environments (and indeed the rest of the enterprise) grow in the coming years.

OpenBSD has been in active development since 1997 and has shown no signs of stopping, so with continued community support the features and security of the operating system could continue to grow.

Chapter 26

Conclusion

In "Part III - Compliance, Financial Regulation and Control" we explored some of the areas of statutory and regulatory compliance, as well as some of the best practice and international standards control sets which are prevalent in today's organisations. OpenBSD's security features (described in "Part II - OpenBSD: Technologies and Features") map cleanly to many of these control areas, and enable organisations to be able to meet requirements and align their controls with their own business processes and functions. In "Part IV - Case Study: Securing Treasury" we saw how OpenBSD could be used in a real-world Treasury environment to provide enhanced security for electronic payments systems and the nodes which feed them, using a defence-in-depth approach of layering OpenBSD's security controls with organisational processes which support them. This project has explored a number of security considerations and challenges in a Financially Sensitive Environment, and the ways in which OpenBSD (and its security features) might be applied to addresses such challenges and work towards the necessary areas of compliance.

As demonstrated in "Part II - OpenBSD: Technologies and Features", OpenBSD has a number of security features, many of which are not present in other operating systems. Additional controls such as memory protection, system call policy enforcement and firewalling with pf have been investigated and their attributes aligned with some of the security challenges faced in business. In Chapter 5 we saw how the pf firewall is able to restrict network traffic defined by policy, which gives organisations the ability to control the traffic that enters or leaves their networks. As discussed, such functionality is available in many systems and offerings, but pfs additional features such as high-availability are available freely without having to pay for commercial licences. In addition, the presence of transparent technologies such as memory protection, bounds checking/enforcing compilers and systrace compilation all add additional layers of security to bring enhanced

control to third-party or even emulated binaries.

I believe that the project objectives have been met, as the project has discussed a number of main security features present within OpenBSD and how they may be used to secure a Treasury environment. Over the course of writing this report, a variety of new issues have been raised over not just the security of OpenBSD, but how this can be realistically applied in a business context, exploring the costs involved and the wider impact of implementation, such as keeping policies current and the need for proper staff training/awareness. For constraining applications, the features of systrace have been tested with a worked example (Chapter 7) and have proven that the system call policies are indeed effective. However, during the course of that testing and writing of this report, a major new flaw was discovered (by another security researcher) in the handling of system calls (Chapter 7.3) which then raised interesting issues about the application of such system call policy technology within the wider realm of enterprise risk management.

There are a number of books available on OpenBSD and its direct application to areas such as firewalls, but I believe this is the first document to focus on the challenges of using OpenBSD in a Financially Sensitive Environment (FSE). This work has contributed the subject area of OpenBSD by seeing how it can be applied in practice to an FSE, including how it maps to some of the compliance and best practice controls which are commonly required for such environments. The scope of this project and the time available has permitted a focused discussion of only some of the security controls against specific control contexts - I believe that there is sufficient scope for far more detailed and expansive investigation into the applicability of OpenBSD to Financially Sensitive Environments in future works.

Bibliography

[1] Balmer, M. Supporting Radio Clocks in OpenBSD (ASIABSD07). http://www.openbsd.org/papers/radio-clocks-asiabsdcon07.pdf. Accessed 2007-07-28.

[2] Barrett, D. J., and Silverman, R. *SSH, The Secure Shell: The Definitive Guide*. O'Reilly, 2001.

[3] Black Viper. Windows 2000 Professional and Server Services Configuration. http://www.blackviper.com/WIN2K/servicecfg.htm. Accessed 2007-07-27.

[4] BSDcertification.org. 2005 BSD Usage Survey. http://www.bsdcertification.org/downloads/pr_20051031_usage_survey_en_en.pdf. Accessed 2007-08-02.

[5] Cabinet Office. Open Source Software: Use Within UK Government. http://www.govtalk.gov.uk/documents/oss_policy_version2.pdf. Accessed 2007-07-04.

[6] Core Security. OpenBSD IPv6 mbufs Remote Kernel Buffer Overflow. http://www.coresecurity.com/?action=item&id=1703. Accessed 2007-08-02.

[7] de Raadt, T. Exploit Mitigation Techniques BSDCAN04. http://www.openbsd.org/papers/auug04/index.html. Accessed 2007-07-04.

[8] de Raadt, T. Exploit Mitigation Techniques PACSEC03. http://www.openbsd.org/papers/pacsec03/e/index.html. Accessed 2007-07-04.

[9] de Raadt, T. Re: defaults for openssh. http://marc.info/?l=openbsd-misc\&m=116223117423784\&w=2. Accessed 2007-07-12.

[10] de Raadt, T. Re: IPFilter licence update. http://marc.info/?l=openbsd-misc&m=99159528204785&w=2. Accessed 2007-07-24.

[11] de Raadt, T. Re: Why were all djb's ports removed? no more qmail? http://marc.info/?l=openbsd-ports\&m=99867670800407\&w=2. Accessed 2007-07-09.

[12] de Raadt, T., and Cranor, C. Opening the Source Repository with Anonymous CVS. http://www.openbsd.org/papers/anoncvs-slides.ps. Accessed 2007-07-04.

[13] de Raadt, T., and Miller, T. C. strlcpy and strlcat - Consistent, Safe, String Copy and Concatenation. http://www.gratisoft.us/todd/papers/strlcpy.html. Accessed 2007-07-04.

[14] Denning, D. E. *Information Warfare and Security*. ACM Press, 1999.

[15] Department for Trade & Industry. DTI Information Security Breaches Survey 2006. http://www.pwc.com/uk/eng/ins-sol/publ/pwc_dti-fullsurveyresults06.pdf. Accessed 2007-07-04.

[16] DMOZ Open Directory Project. Firewall Product Directory. http://www.dmoz.org/Computers/Security/Firewalls/Products/. Accessed 2007-08-29.

[17] Erickson, J. M. *Hacking: The Art of Exploitation*. No Starch Press, 2003.

[18] Financial Services Authority (FSA) United Kingdom. Handbook. http://fsahandbook.info/FSA/html/handbook/. Accessed 2007-07-29.

[19] Fyodor. nmap Network Mapper. http://insecure.org/nmap/. Accessed 2007-06-25.

[20] Garfinkel, S., Spafford, G., and Schwartz, A. *Practical UNIX and Internet Security*, 3rd ed. O'Reilly, 2003.

[21] Gwyne, D. The OpenBSD Culture. http://www.openbsd.org/papers/opencon06-culture.pdf. Accessed 2007-08-01.

[22] IETF Network Working Group. RFC 1918: Address Allocation for Private Internets.
http://www.ietf.org/rfc/rfc1918.txt. Accessed 2007-08-02.

[23] IETF Network Working Group. RFC 2131: Dynamic Host Configuration Protocol.
http://www.ietf.org/rfc/rfc2131.txt. Accessed 2007-08-02.

[24] IETF Network Working Group. RFC 4251: The Secure Shell (SSH) Protocol Architecture.
http://www.ietf.org/rfc/rfc4251.txt. Accessed 2007-08-02.

[25] IETF Network Working Group. RFC 4256: Generic Message Exchange Authentication for the Secure Shell Protocol (SSH).
http://www.ietf.org/rfc/rfc4256.txt. Accessed 2007-08-02.

[26] IETF Secure Shell Working Group. Internet Draft: SSH File Transfer Protocol.
http://tools.ietf.org/html/draft-ietf-secsh-filexfer-13.
Accessed 2007-08-02.

[27] Infosecwriters.com. Sun Solaris 9 Default Configuration Nessus Scan Report.
http://www.infosecwriters.com/projects/osscan/sun9dr.php.
Accessed 2007-07-27.

[28] International Standards Organisation. ISO/IEC 17799 Information Technology - Security Techniques - Code of Practice for Information Security Management (2005). Available from: http://www.bsi-global.com.

[29] International Standards Organisation. ISO/TR 17944 Banking - Security and Other Financial Services - Framework for Security in Financial Systems (2002).

[30] ISACA. Cobit 4.0.
http://www.isaca.org/cobit.htm. Accessed 2007-08-01.

[31] Kelley, D. SOX-in-a-box: One size does not fit all when it comes to compliance.
http://searchsecurity.techtarget.com/tip/0,289483,
sid14_gci1079123,00.html. Accessed 2007-08-27.

[32] Kong, J. *Designing BSD Rootkits: An Introduction to Kernel Hacking*. No Starch Press, 2007.

[33] Lai, R. OpenCVS (BSDCAN07).
http://www.openbsd.org/papers/bsdcan07-cvs/. Accessed 2007-08-04.

[34] Lucas, M. W. *Absolute OpenBSD: UNIX for the Practical Paranoid*. No Starch Press, 2003.

[35] Mason, M. Subversion for CVS Users. `http://osdir.com/Article203.phtml`. Accessed 2007-07-09.

[36] McNab, C. *Network Security Assessment*. O'Reilly, 2004.

[37] Microsoft Corporation. Coporate Home Page. `http://www.microsoft.com/`. Accessed 2007-07-20.

[38] Microsoft Corporation. Internal Firewall Design. `http://www.microsoft.com/technet/security/guidance/networksecurity/secmod155.mspx#E5JAE`. Accessed 2007-08-25.

[39] Microsoft Corporation. Licensing. `http://www.microsoft.com/licensing/default.mspx`. Accessed 2007-07-20.

[40] OpenBSD. Commercial products. `http://www.openbsd.org/products.html`. Accessed 2007-07-02.

[41] OpenBSD. Cryptography. `http://www.openbsd.org/crypto.html`. Accessed 2007-08-01.

[42] OpenBSD. Errata Patches. `http://www.openbsd.org/errata.html`. Accessed 2007-08-06.

[43] OpenBSD. home page. `http://www.openbsd.org/`. Accessed 2007-07-01.

[44] OpenBSD. man page: brconfig(8). `http://www.openbsd.org/cgi-bin/man.cgi?query=brconfig&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html`. Accessed 2007-07-22.

[45] OpenBSD. man page: carp(4). `http://www.openbsd.org/cgi-bin/man.cgi?query=carp\&apropos=0\&sektion=0\&manpath=OpenBSD+Current\&arch=i386\&format=html`. Accessed 2007-07-20.

[46] OpenBSD. man page: hoststated(8). `http://www.openbsd.org/cgi-bin/man.cgi?query=hoststated&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html`. Accessed 2007-07-22.

[47] OpenBSD. man page: pf.conf(5). `http://www.openbsd.org/cgi-bin/man.cgi?query=pf.conf\`

&apropos=0\&sektion=0\&manpath=OpenBSD+Current\&arch=i386\
&format=html. Accessed 2007-07-18.

[48] OpenBSD. man page: pfsync(4).
http://www.openbsd.org/cgi-bin/man.cgi?query=pfsync\
&apropos=0\&sektion=0\&manpath=OpenBSD+Current\&arch=i386\
&format=html. Accessed 2007-07-20.

[49] OpenBSD. man page: release(8).
http://www.openbsd.org/cgi-bin/man.cgi?query=release\
&apropos=0\&sektion=0\&manpath=OpenBSD+Current\&arch=i386\
&format=html. Accessed 2007-07-20.

[50] OpenBSD. man page: securelevel(7).
http://www.openbsd.org/cgi-bin/man.cgi?query=securelevel\
&apropos=0\&sektion=0\&manpath=OpenBSD+Current\&arch=i386\
&format=html. Accessed 2007-07-30.

[51] OpenBSD. man page: sshd_config(5).
http://www.openbsd.org/cgi-bin/man.cgi?query=sshd_config\
&sektion=5\&arch=i386\&apropos=0\&manpath=OpenBSD+Current.
Accessed 2007-07-30.

[52] OpenBSD. man page: syslogd(8).
http://www.openbsd.org/cgi-bin/man.cgi?query=
syslogd&sektion=8&arch=i386&apropos=0&manpath=OpenBSD+
Current. Accessed 2007-07-23.

[53] OpenBSD. man page: systrace(1).
http://www.openbsd.org/cgi-bin/man.cgi?query=systrace\
&apropos=0\&sektion=1\&manpath=OpenBSD+4.1\&arch=i386\
&format=html. Accessed 2007-07-30.

[54] OpenBSD. man page: systrace(4).
http://www.openbsd.org/cgi-bin/man.cgi?query=systrace\
&apropos=0\&sektion=4\&manpath=OpenBSD+4.1\&arch=i386\
&format=html. Accessed 2007-07-30.

[55] OpenBSD. OpenNTPd Project.
http://www.openntpd.org/. Accessed 2007-07-16.

[56] OpenBSD. pf FAQ.
http://www.openbsd.org/faq/pf/index.html. Accessed 2007-08-09.

[57] OpenBSD. pf FAQ: Logging.
http://www.openbsd.org/faq/pf/logging.html. Accessed 2007-08-
09.

[58] OpenBSD. Security.
http://www.openbsd.org/security.html. Accessed 2007-07-02.

[59] OpenBSD. Supported hardware platforms.
http://www.openbsd.org/plat.html. Accessed 2007-07-02.

[60] OpenSSH. Project homepage.
http://www.openssh.org/. Accessed 2007-07-04.

[61] OpenSSH. Systems using OpenSSH.
http://www.openssh.org/users.html. Accessed 2007-07-04.

[62] Ornaghi, A., and Valleri, M. Man in the Middle Attacks Demos.
http://www.blackhat.com/presentations/bh-usa-03/
bh-us-03-ornaghi-valleri.pdf. Accessed 2007-07-29.

[63] Palmer, B. *Secure Architectures with OpenBSD*. Pearson Education Inc., 2004.

[64] PCI Security Standards Council. Payment Card Industry - Data Security Standard v1.1.
https://www.pcisecuritystandards.org/pdfs/pci_dss_v1-1.pdf.
Accessed 2007-08-02.

[65] Peikari, C., and Chuvakin, A. *Security Warrior*. O'Reilly, 2004.

[66] Santana, G. OpenBSD binpatch Project.
http://openbsdbinpatch.sourceforge.net/. Accessed 2007-08-20.

[67] Sauve-Frankel, M. Re: binpatch system.
http://marc.info/?l=openbsd-misc\&m=110607028208153\&w=2.
Accessed 2007-08-20.

[68] Schipper, J. Re: make build — securelevel=2.
http://archives.neohapsis.com/archives/openbsd/2006-01/
1914.html. Accessed 2007-07-06.

[69] Schlyter, J. OpenBSD & BIND 9 cache poisoning.
http://marc.info/?l=openbsd-misc&m=118539211412877&w=2. Accessed 2007-07-28.

[70] SecurityFocus.com. How Not to Respond to a Security Advisory.
http://www.securityfocus.com/columnists/380. Accessed 2007-07-06.

[71] Silberschatz, A., Galvin, P. B., and Gagne, G. *Operating System Concepts*, 7th ed. John Wiley & Sons Inc., 2005.

[72] SLASHDOT.ORG. Remote Exploit Discovered for OpenBSD. http://it.slashdot.org/it/07/03/15/0045207.shtml. Accessed 2007-08-02.

[73] SLASHDOT.ORG. Theo de Raadt Responds (Interview). http://bsd.slashdot.org/article.pl?sid=00/12/11/1455210&mode=thread. Accessed 2007-07-21.

[74] STOLL, C. *The Cuckoo's Egg.* Pan Books, 1990.

[75] SUN MICROSYSTEMS. Solaris 10 Security. http://www.sun.com/software/solaris/security.jsp. Accessed 2007-08-17.

[76] SYSJAIL PROJECT. Sysjail: A Userland Virtualisation System. http://sysjail.bsd.lv/. Accessed 2007-08-12.

[77] THE INSTITUTE OF INTERNAL AUDITORS. Key Strategies for Implementing ISO 27001. http://www.theiia.org/itaudit/index.cfm?catid=21&iid=440. Accessed 2007-07-02.

[78] THOMPSON, K. Reflections on Trusting Trust. *Communication of the Association for Computing Machinery Volume 27, No. 8* (1984). http://www.acm.org/classics/sep95/ Accessed 2007-07-05.

[79] UNDERWOOD, N. HOWTO: Transparent Packet Filtering with OpenBSD. http://ezine.daemonnews.org/200207/transpfobsd.html. Accessed 2007-08-28.

[80] US GOVERNMENT PRINTING OFFICE. Sarbanes Oxley Act of 2002. http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=107_cong_bills\&docid=f:h3763enr.tst.pdf. Accessed 2007-07-15.

[81] WATSON, R. N. M. Exploiting Concurrency Vulnerabilities in System Call Wrappers. http://www.watson.org/~robert/2007woot/2007usenixwoot-exploitingconcurrency.pdf. Accessed 2007-08-13.

[82] WRIGHT, P. M. Time Insecurity and the Network Time Problem (Part 1). http://www.ukcert.org.uk/time_security.html. Accessed 2007-05-30.

Appendix A

Abbreviations

BSD	Berkeley Software Distribution
CARP	Common Address Redundancy Protocol
CVS	Concurrent Versioning System
DHCP	Dynamic Host COnfiguration Protocol
DNS	Domain Name Service
DSS	Data Security Standard
FQDN	Fully Qualified Domain Name
FSE	Financially Sensitive Environment
FTP	File Transfer Protocol
FTPS	FTP Secure (with SSL)
GPL	GNU Public Licence
GPS	Global Positioning System
IPSec	Internet Protocol Security
ISC	Internet Software Consortium
ISO	International Standards Organisation
ITIL	Information technology Infrastructure Library
LAN	Local Area Network
NAT	Network Address Translation
NTP	Network Time Protocol
PCI	Payment Card Industry
PoC	Proof-of-Concept
QoS	Quality of Service
RAID	Redundant Array of Inexpensive Disks
SCP	Secure CoPy
SFTP	Secure FTP (with SSH)
SSH	Secure SHell
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WAN	Wide Area Network